MINISTRY OF EDUCATION AND SCIENCE OF THE REPUBLIC OF KAZAKHSTAN

JSC "Kazakh-British Technical University"
Faculty of Information Technologies

"ADMITTED TO DEFENCE"
Chair of CE Department: *Lyazzat B. Atymtayeva*

Writer of dissertation: *Vadim V. Kotov*

MASTER'S DISSERTATION

6M070300 – "Information Systems" specialty

Theme: **"Game Mechanics for Stimulating High Performance of Project Participants"**

Scientific supervisor

Timur F. Umarov,
Ph.D. Computer Science,
Associate professor

Almaty, 2013

MINISTRY OF EDUCATION AND SCIENCE OF THE REPUBLIC OF KAZAKHSTAN

JSC "Kazakh-British Technical University"
Faculty of Information Technologies

"APPROVED BY"
Chair of CE Department
Lyazzat B. Atymtayeva,
Doctor of physical and mathematical sciences,
Professor

"____" _____ 2013

ASSIGNMENT FOR MASTER'S DISSERTATION

V. Kotov

6M070300 – Information Systems

Theme: "Game Mechanics for Stimulating High Performance of Project Participants"

Objectives: Explore modern project management methodologies and gamification theory in order to understand elements affecting performance of project participants and provide guidelines in order to increase their effectiveness.

Source data:

- Sketch (Bohemian Coding)
- Processing (http://processing.org/)

Scientific supervisor

Timur F. Umarov,
Ph.D. Computer Science,
Associate professor

"____" _____ 2013

Almaty, 2013

MINISTRY OF EDUCATION AND SCIENCE OF THE REPUBLIC OF KAZAKHSTAN

JSC "Kazakh-British Technical University"
Department of Computer Engineering

SCIENTIFIC PUBLICATIONS

by Information Systems M.Sc. programme student Vadim Kotov

No.	Title	Publisher	Pages	Co-authors

Author

Vadim Kotov

Scientific secretary

MINISTRY OF EDUCATION AND SCIENCE OF THE REPUBLIC OF KAZAKHSTAN

JSC "Kazakh-British Technical University"
Department of Computer Engineering

REVIEW

by M.Sc. thesis supervisor
Timur Umarov

on **"Game Mechanics for Stimulating High Performance of Project Participants"**

6M070300 – Information Systems

Thesis supervisor
Ph. D.,

_____

Timur Umarov

MINISTRY OF EDUCATION AND SCIENCE OF THE REPUBLIC OF KAZAKHSTAN

Opponent's review of the Master's Dissertation

"Game Mechanics for Stimulating High Performance of Project Participants"

Kazakh-British Technical University
6M070300 – Information Systems
Vadim V. Kotov

Master's Dissertation of Vadim Kotov explores several project management methodologies (known as iterative) and gamification theory in order to understand elements affecting performance of project participants and provide guidelines in order to increase their effectiveness.

This Dissertation introduces a complex overview of causes of emerging iterative methodologies, motivation management and the concept of "Flow". The resulting application prototype incorporates a very different approach and focuses on feedback.

Dissertation consists of 5 chapters, presenting the necessary background combination of Project Management methodologies and psychological mechanisms behind motivation.

In addition I wish to say that it is a very solid work. Results from the research could be used to stimulate not only project participants' performance, but also personal performance. The dissertation is a good starting point for many further researches on system thinking in project management.

I recommend to evaluate the dissertation of Vadim V. Kotov as "excellent" and award him with a deserved qualification "MSc in Information Systems" by specialty 6M070300.

Opponent:
Maksat Maratov (M.Sc. in Information Systems)
Lecturer, Department of Information Technology
International Information Technology University

Abstract

Master dissertation consists of 77 of the typewritten text, xx figures, xx tables and xx reference materials. The following keywords and terms are used in the paper:

GAMIFICATION, GAME MECHANICS, FLOW, AGILE, SCRUM, RAPID APPLICATION DEVELOPMENT, TEST-DRIVEN DEVELOPMENT, FEATURE DRIVEN DEVELOPMENT, LEAN, PROGRAM EVALUATION AND REVIEW TECHNIQUE

Dissertation contains an overview of modern iterative project management techniques, team and personal motivation strategies, gamification methods and productivity mobile applications.

The objective was to explore modern project management methodologies and gamification theory in order to understand elements affecting performance of project participants and provide guidelines in order to increase their effectiveness.

The result of the research is a methodology, based on causes of emerging iterative methodologies, motivation management and the concept of "Flow" and an application prototype, that uses concept of "Feedback" in order to visualise limited resources, such as time and focus.

Contents

List of Figures

List of Tables

Introduction

Efficiency and effectiveness are very important in any project. They determine the speed of moving towards goals and required resources. These concepts are affected by a number of various reasons including motivation of team members, project management methodology, ability to focus, etc.

There is a big number of different methodologies, motivational systems, personal efficiency / effectiveness trainings, etc. But in the study, conducted by Dean Spitzer, as many as 50 percent of workers said they only put enough effort into their work to hold onto their jobs. And 84 percent said they could work better - if they wanted to. [1]

So the problem is clear: there is no complex solution, because of misunderstanding of how and why do parts of the "project" system relate to each other.

Chapter 1 contains an overview of modern iterative project management techniques.

Chapter 2 looks into team-motivation strategies, gamification and Mihaly Csikszentmihalyi's "Flow" theory.

The following chapters will consider market leaders (chapter 3) in productivity mobile applications. Takes a look at team-wide motivation techniques (chapter 4) and suggests a new approach in task list description.

The last chapter contains the prototype of resulting application.

1. Project management techniques for small teams and startups

In the past project management was different. Not only because the formal discipline was mostly applied to the large projects lasting several years and costing millions of dollars, but also used a different approach, evolved from ancient military regimes, where relatively few people directed large number of others [2].

Project definition, as a temporary endeavour with a defined beginning and end [3], undertaken to meet unique goals and objectives [4] extends to the simple premise: everything is a project. For example important presentation or "career development" project, or even employee development (each employee represents a single "project" in which it is required to keep track of performance and plan to help him or her develop). Pursuing goals makes project management techniques essential to get things done, and especially important for personal ones.

Downscaling project management to apply to the personal projects or small businesses required a different approach. In comparison to traditional "waterfall" model, which is inflexible and can potentially lead to a harmful consequences, a number of new methods emerged.

1.1 Agile methods

A group of such methods, called "Agile" are a reaction to traditional ways of developing software and acknowledge the "need for an alternative to documentation driven, heavyweight software development processes".

The core of these methods consists of adaptive planning, evolutionary development and delivery, an iterative approach, and rapid and flexible response to change.

In 2001 the Manifesto for Agile Software Development [5] was published to define the approach now known as agile software development.

Some of the manifesto's authors formed the Agile Alliance (http://www.agilealliance.org/), a nonprofit organization that promotes software de-

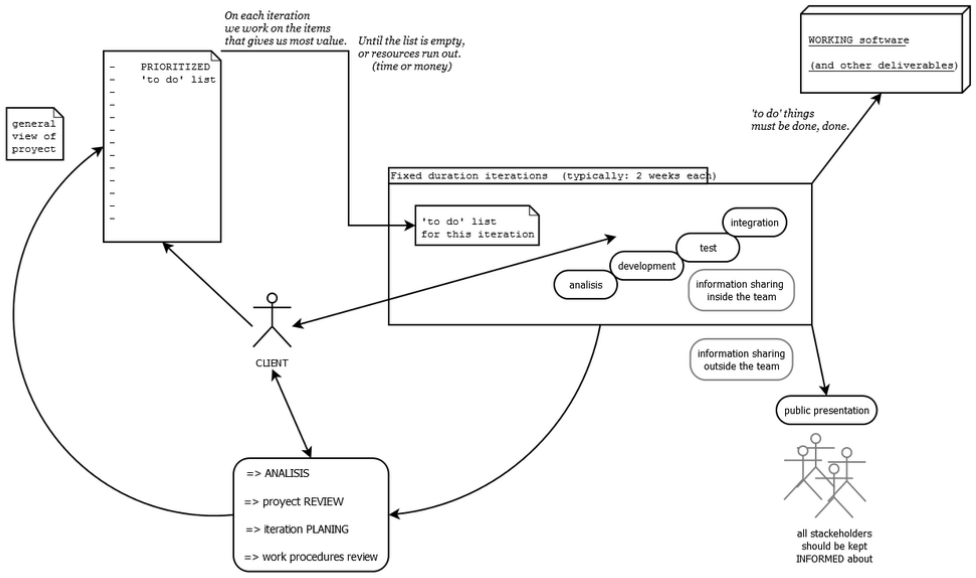

Figure 1.1: Generic agile diagram

velopment according to the manifesto's principles.

The Agile Manifesto:

> We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:
>
> - Individuals and interactions over processes and tools
> - Working software over comprehensive documentation
> - Customer collaboration over contract negotiation
> - Responding to change over following a plan
>
> That is, while there is value in the items on the right, we value the items on the left more.

Meaning, that self-organisation and motivation are important, working software will be more useful and welcome as a presentation for clients in meetings, collaboration leads to a better product, effectively tailored to the customer needs and quick changes allow higher quality of product.

According to Kent Beck,[6] the Agile Manifesto is based on twelve principles:

1. Customer satisfaction by rapid delivery of useful software
2. Welcome changing requirements, even late in development
3. Working software is delivered frequently (weeks rather than months)

4. Working software is the principal measure of progress
5. Sustainable development, able to maintain a constant pace
6. Close, daily cooperation between business people and developers
7. Face-to-face conversation is the best form of communication (co-location)
8. Projects are built around motivated individuals, who should be trusted
9. Continuous attention to technical excellence and good design
10. Simplicitythe art of maximizing the amount of work not doneis essential
11. Self-organizing teams
12. Regular adaptation to changing circumstances

Agile development is popular in a certain types of environment, including small teams. However a number of thing may negatively impact the success of an agile project:

1. Large-scale development efforts (>20 developers), though scaling strategies [7] and evidence of some large projects [8] have been described.
2. Distributed development efforts (non-colocated teams). Still there are examples of successful companies (37 Signals).
3. Forcing an agile process on a development team
4. Mission-critical systems where failure is not an option at any cost (e.g. software for air traffic control).

Agile methods have been extensively used for development of software products and some of them use certain characteristics of software, such as object technologies. However, these techniques can be applied to the development of non-software products, such as computers, motor vehicles, medical devices, food, and clothing.

1.2 Scrum

One of the implementations of Agile philosophy is Scrum.

Authors describe Scrum, [9] as a framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value.

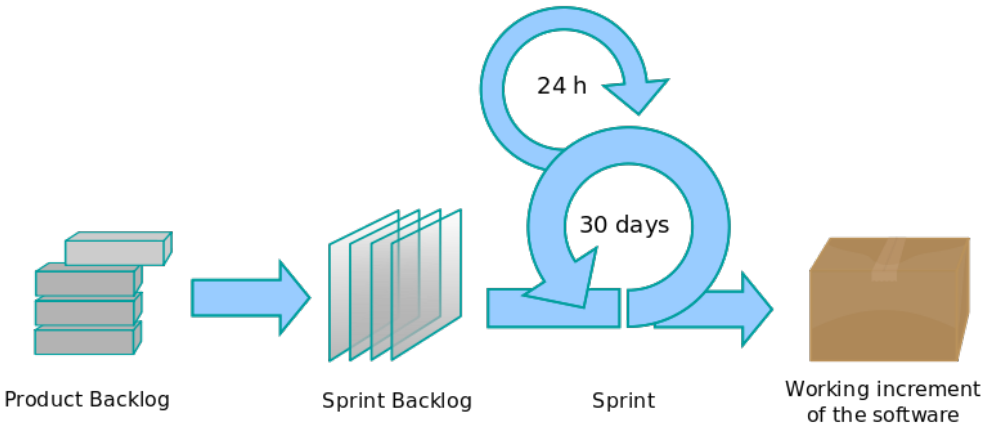

Figure 1.2: Scrum process

1.2.1 The Scrum team

The Scrum Team consists of a Product Owner, the Development Team, and a Scrum Master. Scrum Teams are self-organising and cross-functional. Self-organising teams choose how best to accomplish their work, rather than being directed by others outside the team. Cross-functional teams have all competencies needed to accomplish the work without depending on others not part of the team. The team model in Scrum is designed to optimise flexibility, creativity, and productivity.

Scrum Teams deliver products iteratively and incrementally, maximising opportunities for feedback. Incremental deliveries of "Done" product ensure a potentially useful version of working product is always available.

1.2.2 Scrum events

Prescribed events are used in Scrum to create regularity and to minimise the need for meetings not defined in Scrum. Scrum uses time-boxed events, such that every event has a maximum duration. This ensures an appropriate amount of time is spent planning without allowing waste in the planning process.

Other than the Sprint itself, which is a container for all other events, each event in Scrum is a formal opportunity to inspect and adapt something. These events are specifically designed to enable critical transparency and inspection. Failure to include any of these events results in reduced transparency and is a lost opportunity to inspect and adapt.

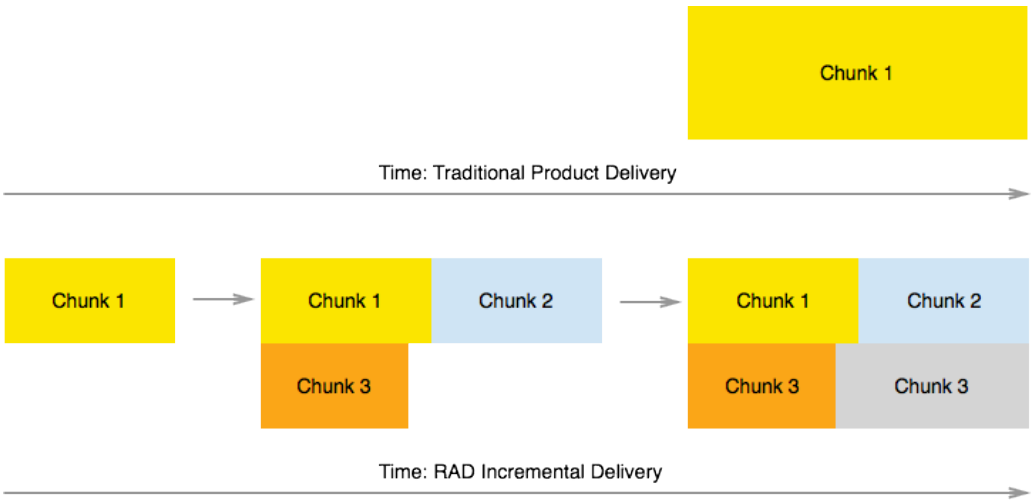

Figure 1.3: RAD incremental delivery

1.2.3 Scrum artifacts

Scrums artifacts represent work or value in various ways that are useful in providing transparency and opportunities for inspection and adaptation. Artifacts defined by Scrum are specifically designed to maximise transparency of key information needed to ensure Scrum Teams are successful in delivering a "Done" Increment.

1.3 RAD – Rapid Application Development

RAD is an integrated set of techniques, guidelines and tools that facilitate deploying a customer's software needs within a short period of time. This predefined timeframe is called a "timebox". The software product evolves during the RAD development process based on continued customer feedback. In addition, the whole software product is not delivered at once, but is delivered in pieces by order of business importance. [10]

The RAD process defies a linear definition of steps carried out in a sequence. [10]

1.3.1 Phases of RAD

1. Requirements Planning phase – combines elements of the system planning and systems analysis. During this stage, a definition of the project scope is completed along with some preliminary data/process analysis, risk assessment and estimating.

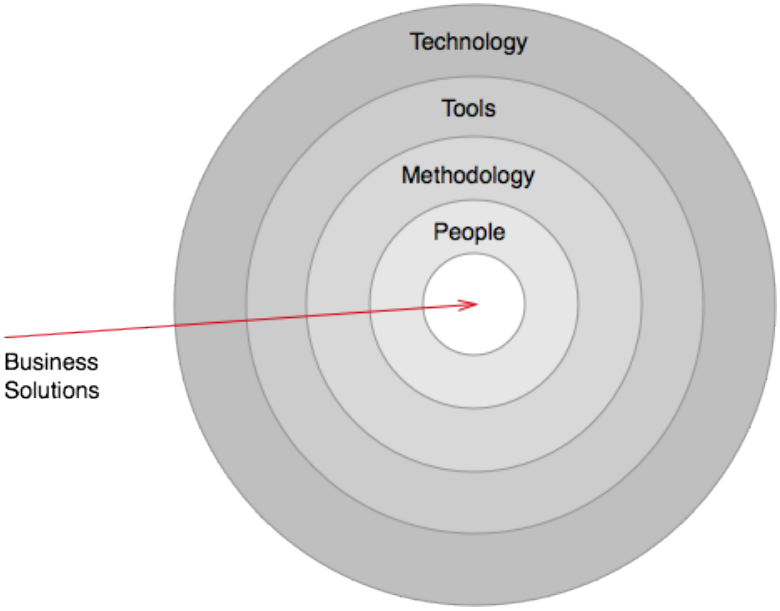

Figure 1.4: Bull's eye for RAD, it should he the business problem, not the technology.

2. User design phase – during this phase, users interact with systems analysts and develop models and prototypes that represent all system processes, inputs, and outputs. The RAD groups or subgroups typically use a combination of Joint Application Development (JAD) techniques and CASE tools to translate user needs into working models. User Design is a continuous interactive process that allows users to understand, modify, and eventually approve a working model of the system that meets their needs.

3. Construction phase – focuses on program and application development task similar to the SDLC. In RAD, however, users continue to participate and can still suggest changes or improvements as actual screens or reports are developed. Its tasks are programming and application development, coding, unit-integration and system testing.

4. Cutover phase – resembles the final tasks in the SDLC implementation phase, including data conversion, testing, changeover to the new system, and user training. Compared with traditional methods, the entire process is compressed. As a result, the new system is built, delivered, and placed in operation much sooner.

In addition to using such techniques as timeboxing, chunking and customer-driven product delivery, RAD is based on the premise that software development

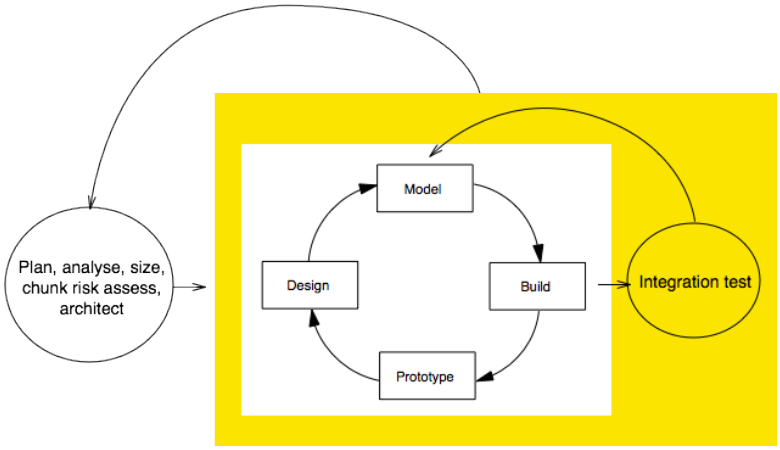

Figure 1.5: RAD timeboxing for each "chunk". It forces project team to have a market product orientation

is a discovery process.

1.4 TDD – Test-Driven Development

Test-driven development (TDD) is a software development process that relies on the repetition of a very short development cycle: first the developer writes an (initially failing) automated test case that defines a desired improvement or new function, then produces the minimum amount of code to pass that test, and finally refactors the new code to acceptable standards. Kent Beck, who is credited with having developed or 'rediscovered' the technique, stated in 2003 that TDD encourages simple designs and inspires confidence.

1.4.1 TDD Cycle

The TDD techniques is particularly interesting in terms of feedback, which helps to fail fast, be flexible and get a feeling of moving towards a goal. This is reached by using the following cycle.

Add a test

In test-driven development, each new feature begins with writing a test. This test must inevitably fail because it is written before the feature has been implemented. (If it does not fail, then either the proposed "new" feature already exists or the test is defective.) To write a test, the developer must clearly understand the feature's specification and requirements. The developer can

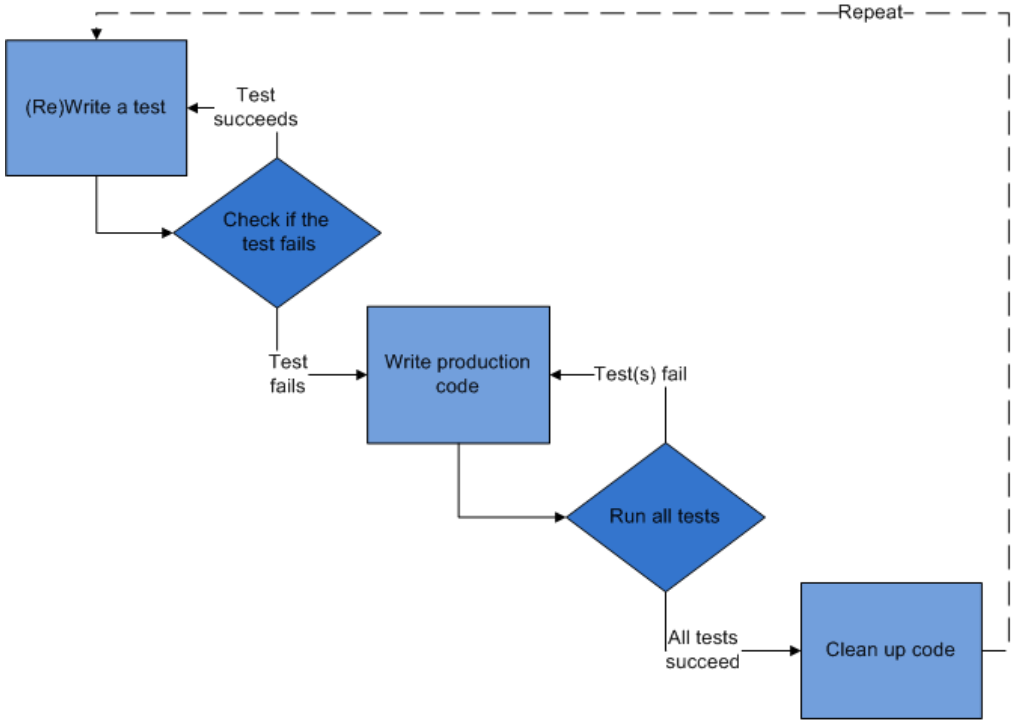

Figure 1.6: Test-Driven Development Cycle

accomplish this through use cases and user stories to cover the requirements and exception conditions, and can write the test in whatever testing framework is appropriate to the software environment. This could also be a modification of an existing test. This is a differentiating feature of test-driven development versus writing unit tests after the code is written: it makes the developer focus on the requirements before writing the code, a subtle but important difference.

Run all tests and see if the new one fails

This validates that the test harness is working correctly and that the new test does not mistakenly pass without requiring any new code. This step also tests the test itself, in the negative: it rules out the possibility that the new test always passes, and therefore is worthless. The new test should also fail for the expected reason. This increases confidence (though does not guarantee) that it is testing the right thing, and passes only in intended cases.

Write some code

The next step is to write some code that causes the test to pass. The new code written at this stage is not perfect, and may, for example, pass the test in

an inelegant way. That is acceptable because later steps improve and hone it.

At this point, the only purpose of the written code is to pass the test; no further (and therefore untested) functionality should be predicted and 'allowed for' at any stage.

Run the automated tests and see them succeed

If all test cases now pass, the programmer can be confident that the code meets all the tested requirements. This is a good point from which to begin the final step of the cycle.

Refactor code

Now the code can be cleaned up as necessary. By re-running the test cases, the developer can be confident that code refactoring is not damaging any existing functionality. The concept of removing duplication is an important aspect of any software design. In this case, however, it also applies to removing any duplication between the test code and the production codefor example magic numbers or strings repeated in both to make the test pass in step 3.

Repeat

Starting with another new test, the cycle is then repeated to push forward the functionality. The size of the steps should always be small, with as few as 1 to 10 edits between each test run. If new code does not rapidly satisfy a new test, or other tests fail unexpectedly, the programmer should undo or revert in preference to excessive debugging. Continuous integration helps by providing revertible checkpoints. When using external libraries it is important not to make increments that are so small as to be effectively merely testing the library itself, unless there is some reason to believe that the library is buggy or is not sufficiently feature-complete to serve all the needs of the main program being written.

1.5 FDD – Feature-Driven Development

FDD allows to manage projects at a very high level and apply other methodologies (such as TDD) at a lower level of abstraction. It is an iterative and incremental software development process, which primary focus is on being able to set estimates and schedules and to report on the status of a project, or its part.

1.5.1 Phases of FDD

FDD consists of five basic activities. For accurate state reporting and keeping track of the software development project, milestones that mark the progress made on each feature are defined.

Develop overall model

The project started with a high-level walkthrough of the scope of the system and its context. Next, detailed domain walkthroughs were held for each modelling area. In support of each domain, walkthrough models were then composed by small groups, which were presented for peer review and discussion. One of the proposed models, or a merge of them, was selected which became the model for that particular domain area. Domain area models were merged into an overall model, and the overall model shape was adjusted along the way.

Build feature list

The knowledge that was gathered during the initial modelling was used to identify a list of features. This was done by functionally decomposing the domain into subject areas. Subject areas each contain business activities, the steps within each business activity formed the categorised feature list. Features in this respect were small pieces of client-valued functions expressed in the form "¡action¿ ¡result¿ ¡object¿", for example: 'Calculate the total of a sale' or 'Validate the password of a user'. Features should not take more than two weeks to complete, else they should be broken down into smaller pieces.

Plan by feature

After the feature list had been completed, the next step was to produce the development plan. Class ownership has been done by ordering and assigning features (or feature sets) as classes to chief programmers.

Design by feature

A design package was produced for each feature. A chief programmer selected a small group of features that are to be developed within two weeks. Together with the corresponding class owners, the chief programmer worked out detailed sequence diagrams for each feature and refines the overall model. Next, the class and method prologues are written and finally a design inspection is held.

Build by feature

After a successful design inspection a per feature activity to produce a completed client-valued function (feature) is being produced. The class owners develop the actual code for their classes. After a unit test and a successful code inspection, the completed feature is promoted to the main build.

1.6 Lean

Lean development is based on traditional lean principles, which are derived from the Japanese manufacturing industry. Generally "Lean" is a production practice that considers the expenditure of resources for any goal other than the creation of value for the end customer to be wasteful, and thus a target for elimination. Working from the perspective of the customer who consumes a product or service, "value" is defined as any action or process that a customer would be willing to pay for.

Lean development can be summarised by seven principles, very close in concept to lean manufacturing principles:

1. Eliminate waste
2. Amplify learning
3. Decide as late as possible

4. Deliver as fast as possible
5. Empower the team
6. Build integrity in
7. See the whole

1.6.1 Lean manufacturing principles

Eliminate waste

Everything not adding value to the customer is considered to be waste (muda). This includes:

1. unnecessary code and functionality
2. delay in the software development process
3. unclear requirements
4. insufficient testing, leading to avoidable process repetition
5. bureaucracy
6. slow internal communication

In order to be able to eliminate waste, one should be able to recognise it. If some activity could be bypassed or the result could be achieved without it, it is waste. Partially done coding eventually abandoned during the development process is waste. Extra processes and features not often used by customers are waste. Waiting for other activities, teams, processes is waste. Defects and lower quality are waste. Managerial overhead not producing real value is waste. A value stream mapping technique is used to distinguish and recognise waste. The second step is to point out sources of waste and eliminate them. The same should be done iteratively until even essential-seeming processes and procedures are liquidated.

Amplify learning

Software development is a continuous learning process with the additional challenge of development teams and end product sizes. The best approach for improving a software development environment is to amplify learning. The accumulation of defects should be prevented by running tests as soon as the code is written. Instead of adding more documentation or detailed planning, different ideas could be tried by writing code and building. The process of

user requirements gathering could be simplified by presenting screens to the end-users and getting their input.

The learning process is sped up by usage of short iteration cycles – each one coupled with refactoring and integration testing. Increasing feedback via short feedback sessions with customers helps when determining the current phase of development and adjusting efforts for future improvements. During those short sessions both customer representatives and the development team learn more about the domain problem and figure out possible solutions for further development. Thus the customers better understand their needs, based on the existing result of development efforts, and the developers learn how to better satisfy those needs. Another idea in the communication and learning process with a customer is set-based development – this concentrates on communicating the constraints of the future solution and not the possible solutions, thus promoting the birth of the solution via dialogue with the customer.

Decide as late as possible

As software development is always associated with some uncertainty, better results should be achieved with an options-based approach, delaying decisions as much as possible until they can be made based on facts and not on uncertain assumptions and predictions. The more complex a system is, the more capacity for change should be built into it, thus enabling the delay of important and crucial commitments. The iterative approach promotes this principle – the ability to adapt to changes and correct mistakes, which might be very costly if discovered after the release of the system.

An agile software development approach can move the building of options earlier for customers, thus delaying certain crucial decisions until customers have realised their needs better. This also allows later adaptation to changes and the prevention of costly earlier technology-bounded decisions. This does not mean that no planning should be involved – on the contrary, planning activities should be concentrated on the different options and adapting to the current situation, as well as clarifying confusing situations by establishing patterns for rapid action. Evaluating different options is effective as soon as it is realised that they are not free, but provide the needed flexibility for late decision making.

Deliver as fast as possible

In the era of rapid technology evolution, it is not the biggest that survives, but the fastest. The sooner the end product is delivered without considerable defect, the sooner feedback can be received, and incorporated into the next iteration. The shorter the iterations, the better the learning and communication within the team. Without speed, decisions cannot be delayed. Speed assures the fulfilling of the customer's present needs and not what they required yesterday. This gives them the opportunity to delay making up their minds about what they really require until they gain better knowledge. Customers value rapid delivery of a quality product.

The just-in-time production ideology could be applied to software development, recognising its specific requirements and environment. This is achieved by presenting the needed result and letting the team organise itself and divide the tasks for accomplishing the needed result for a specific iteration. At the beginning, the customer provides the needed input. This could be simply presented in small cards or stories – the developers estimate the time needed for the implementation of each card. Thus the work organisation changes into self-pulling system – each morning during a stand-up meeting, each member of the team reviews what has been done yesterday, what is to be done today and tomorrow, and prompts for any inputs needed from colleagues or the customer. This requires transparency of the process, which is also beneficial for team communication. Another key idea in Toyota's Product Development System is set-based design. If a new brake system is needed for a car, for example, three teams may design solutions to the same problem. Each team learns about the problem space and designs a potential solution. As a solution is deemed unreasonable, it is cut. At the end of a period, the surviving designs are compared and one is chosen, perhaps with some modifications based on learning from the others – a great example of deferring commitment until the last possible moment. Software decisions could also benefit from this practice to minimise the risk brought on by big up-front design.

Empower the team

There has been a traditional belief in most businesses about the decision-making in the organisation – the managers tell the workers how to do their own job. In a Work-Out technique, the roles are turned – the managers are taught how to listen to the developers, so they can explain better what actions might be taken, as well as provide suggestions for improvements. The lean approach favours the aphorism "find good people and let them do their own job," encouraging progress, catching errors, and removing impediments, but not micro-managing.

Another mistaken belief has been the consideration of people as resources. People might be resources from the point of view of a statistical data sheet, but in software development, as well as any organisational business, people do need something more than just the list of tasks and the assurance that they will not be disturbed during the completion of the tasks. People need motivation and a higher purpose to work for – purpose within the reachable reality, with the assurance that the team might choose its own commitments. The developers should be given access to the customer; the team leader should provide support and help in difficult situations, as well as ensure that skepticism does not ruin the teams spirit.

Build integrity in

The customer needs to have an overall experience of the System – this is the so-called perceived integrity: how it is being advertised, delivered, deployed, accessed, how intuitive its use is, price and how well it solves problems.

Conceptual integrity means that the systems separate components work well together as a whole with balance between flexibility, maintainability, efficiency, and responsiveness. This could be achieved by understanding the problem domain and solving it at the same time, not sequentially. The needed information is received in small batch pieces – not in one vast chunk with preferable face-to-face communication and not any written documentation. The information flow should be constant in both directions – from customer to developers and back, thus avoiding the large stressful amount of information after long development in isolation.

One of the healthy ways towards integral architecture is refactoring. As more features are added to the original code base, the harder it becomes to add further improvements. Refactoring is about keeping simplicity, clarity, minimum amount of features in the code. Repetitions in the code are signs for bad code designs and should be avoided. The complete and automated building process should be accompanied by a complete and automated suite of developer and customer tests, having the same versioning, synchronization and semantics as the current state of the System. At the end the integrity should be verified with thorough testing, thus ensuring the System does what the customer expects it to. Automated tests are also considered part of the production process, and therefore if they do not add value they should be considered waste. Automated testing should not be a goal, but rather a means to an end, specifically the reduction of defects.

See the whole

Software systems nowadays are not simply the sum of their parts, but also the product of their interactions. Defects in software tend to accumulate during the development process – by decomposing the big tasks into smaller tasks, and by standardising different stages of development, the root causes of defects should be found and eliminated. The larger the system, the more organisations that are involved in its development and the more parts are developed by different teams, the greater the importance of having well defined relationships between different vendors, in order to produce a system with smoothly interacting components. During a longer period of development, a stronger subcontractor network is far more beneficial than short-term profit optimising, which does not enable win-win relationships.

1.7 Summarising agile, RAD and lean

There is a common feature in all of the considered methodologies. According to table 1.1, that is constructing a continuous feedback, which allows early failure and indicates whether a project goal is correct (or it should be adjusted) and a team progress. Each methodology provides its own level of abstraction and suggests tools to reduce chaos and uncertainty.

Comparing iterative methodologies to traditional waterfall model it is possible to say, that they appeared because of emergent projects with elements changing fast (goals, business processes, markets, etc.). Nonetheless traditional project management provides tools to cope with uncertainties, called PERT (section 1.7.1).

Ability to get instant feedback and adjust goals is defined by a natural human attribute as pursuing order in consciousness. This claim will be considered in section 2.4 on page 46.

Table 1.1: Pros and cons of iterative methodics

Name	**Pros**	**Cons**
Agile	Minimises feature creep by developing in short intervals resulting in miniature software projects and releasing the product in mini-increments.	Short iteration may add too little functionality, leading to significant delays in final iterations. Since Agile emphasises real-time communication (preferably face-to-face), using it is problematic for large multi-team distributed system development. Agile methods produce very little written documentation and require a significant amount of post-project documentation.
Lean	Creates minimalist solutions and delivers less functionality earlier; per the policy that 80% today is better than 100% tomorrow.	Product may lose its competitive edge because of insufficient core functionality and may exhibit poor overall quality.

Table 1.1: Pros and cons of iterative methodics

Name	**Pros**	**Cons**
RAD	Promotes strong collaborative atmosphere and dynamic gathering of requirements. Business owner actively participates in prototyping, writing test cases and performing unit testing.	Dependence on strong cohesive teams and individual commitment to the project. Decision-making relies on the feature functionality team and a communal decision-making process with lesser engineering authority.
Scrum	Agile framework. Improved productivity in teams previously paralysed by heavy "process", ability to prioritise work, use of backlog for completing items in a series of short iterations or sprints, daily measured progress and communications.	Reliance on facilitation by a master who may lack the political skills to remove impediments and deliver the sprint goal. Due to reliance on self-organising teams and rejection of traditional centralised "process control", internal power struggles can paralyse a team.

1.7.1 Program Evaluation and Review Technique

Let's also consider traditional "waterfall" methodology statistical tool, designed to analyse and represent the tasks involved in completing a given project called PERT (Program Evaluation and Review Technique).

PERT was developed for the U.S. Navy Special Projects Office in 1957 to support the U.S. Navy's Polaris nuclear submarine project [11]. It was able to incorporate uncertainty by making it possible to schedule a project while not knowing precisely the details and durations of all the activities. It is more of an event-oriented technique rather than start- and completion-oriented, and is used more in projects where time is the major factor rather than cost. It

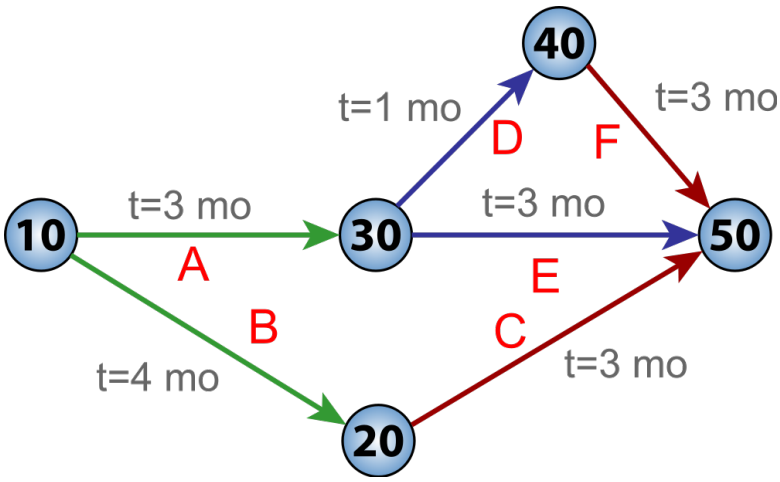

Figure 1.7: PERT Chart

is applied to very large-scale, one-time, complex, non-routine infrastructure and Research and Development projects. An example of this was for the 1968 Winter Olympics in Grenoble which applied PERT from 1965 until the opening of the 1968 Games.

Conventions

- A PERT chart is a tool that facilitates decision making. The first draft of a PERT chart will number its events sequentially in 10s (10, 20, 30, etc.) to allow the later insertion of additional events.
- Two consecutive events in a PERT chart are linked by activities, which are conventionally represented as arrows.
- The events are presented in a logical sequence and no activity can commence until its immediately preceding event is completed.
- The planner decides which milestones should be PERT events and also decides their "proper" sequence.
- A PERT chart may have multiple pages with many sub-tasks.

Advantages

- PERT chart explicitly defines and makes visible dependencies (precedence relationships) between the work breakdown structure (commonly WBS) elements
- PERT facilitates identification of the critical path and makes this visible

- PERT facilitates identification of early start, late start, and slack for each activity,
- PERT provides for potentially reduced project duration due to better understanding of dependencies leading to improved overlapping of activities and tasks where feasible.
- The large amount of project data can be organised and presented in diagram for use in decision making.

Disadvantages

- There can be potentially hundreds or thousands of activities and individual dependency relationships
- PERT is not easily scalable for smaller projects
- The network charts tend to be large and unwieldy requiring several pages to print and requiring special size paper
- The lack of a timeframe on most PERT/CPM charts makes it harder to show status although colours can help (e.g., specific colour for completed nodes)
- When the PERT/CPM charts become unwieldy, they are no longer used to manage the project.

Uncertainty in project scheduling

A real-life project will never execute exactly as it was planned due to uncertainty. It can be ambiguity resulting from subjective estimates that are prone to human errors or it can be variability arising from unexpected events or risks (which can drastically affect motivation of project participants, as the new information is in conflict with personal structure of goals). The main reason that PERT may provide inaccurate information about the project completion time is due to this schedule uncertainty. This inaccuracy is large enough to render such estimates as not helpful.

One possibility to maximise solution robustness is to include safety in the baseline schedule in order to absorb the anticipated disruptions. This is called proactive scheduling. A pure proactive scheduling is a utopia, incorporating safety in a baseline schedule that allows to cope with every possible disruption would lead to a baseline schedule with a very large make-span. A second

approach, reactive scheduling, consists of defining a procedure to react to disruptions that cannot be absorbed by the baseline schedule.

Another approach incorporates Monte-Carlo analysis and simulates potential effects of schedule shifts.

1.7.2 Tracking status of a project

So, there is a technique to collect feedback in a large-scale projects. As any project could be considered as a system, it has different number of parts. As complexity of a system increases exponentially /citeoconnor, PERT tries to define relations and predict possible system states.

Use of iterative methods simplifies the system, and makes cause-and-effect relation clear.

Now team working on a project is able to collect necessary feedback in order to understand the state of a system. Thus it is possible to measure effectiveness (and make necessary adjustments to processes). But team members should be motivated in order to perform best.

1.8 "Action Method" application and concept behind

This section describes an interesting technique, created by Behance company [12]. It will be further used to construct an effectiveness guide.

The Action Method begins with a simple premise: everything is a project. Most creative people struggle to make progress in all of their projects, with the greatest challenge being the sheer number of projects before a person! But once everything has been classified as a project, it is possible to break each one down into its primary components: Action Steps, References, and Backburner Items.

Every project in life can be reduced into these three primary components.

1.8.1 Primary components of "Action Method"

Action Steps

Action steps are the specific, concrete tasks: redraft and send the memo, post the blog entry, pay the electricity bill, etc.

References

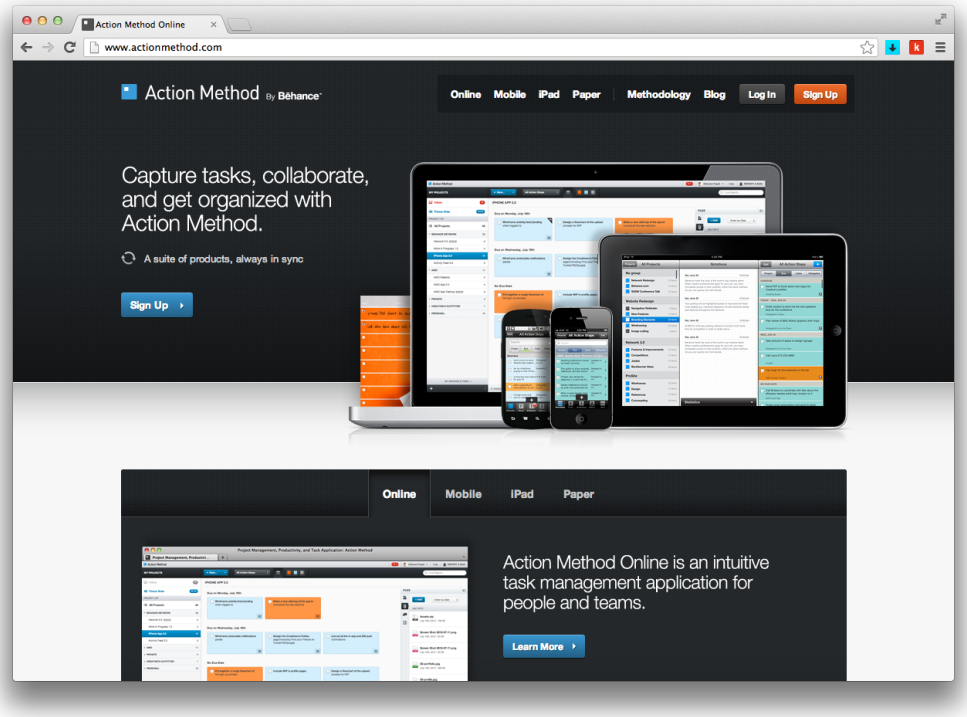

Figure 1.8: Action Method Landing Page by Behance

References are any project-related handouts, sketches, notes, meeting minutes, manuals, websites, or ongoing discussions to refer back to. It is important to note that references are not actionablethey are simply there for reference when focusing on any particular project.

Backburner Items

Backburner items – things that are not actionable now but may be someday. Perhaps it is an idea for a client for which there is no budget yet. Or maybe it is something you intend to do in a particular project at an unforeseen time in the future.

Every project in life can be reduced into these three primary components.

Lets consider a sample project for a client. Assume a folder with that clients name on it. Inside the folder there is a lot of References – a copy of the contract, notes from meetings, and background information on the client. The Action Steps (to-do list) could be written as a list, attached to the front of the folder. On a sheet stapled to the inside back cover of the folder, Backburner list could keep track of the non-actionable ideas that come up while working on the project.

With this hypothetical folder in mind, it is possible to imagine that the majority of focus would be on the Action Steps visible on the front cover. These

Action Steps are always in plain view, while other parts accessible during the review phase.

Personal projects can also be broken down into the same three elements. The Action Method starts by considering everything with a project lens and then breaking it down.

1.8.2 Action Steps in detail

Action Steps are the most important components of projects. The actual outcome of any idea is dependent on the Actions Steps that are captured and then completed by you or delegated to someone else. Action Steps are to be revered and treated as sacred in any project. The more clear and concrete an Action Step is, the less friction a person will encounter trying to do it. If an Action Step is vague or complicated, a person will probably skip over it to others on the list that are more straightforward.

To avoid this, it is required to start each Action Step with a verb:

- Call programmer to discuss...
- Install new software for...
- Research the possibility of...
- Mock up a sample of the...
- Update XYZ document for...

Verbs help to pull into Action Steps at first glance, efficiently indicating what type of action is required. For similar reasons, Action Steps should be kept short.

Ideas dont reveal themselves only in meetings, and neither should Action Steps.

An unowned Action Step will never be taken.

Every Action Step must be owned by a single person. While some Action Steps may involve the input of different people, accountability must reside in one individuals hands. Some people who lead teams or have assistants will capture Action Steps and delegate them to others. However, Action Step must still be owned by the person ultimately responsible.

Every Action Step must be owned by a single person.

The reason comes down to accountability. The practice of simply emailing someone a task to complete does not provide any assurance that it will be

completed. For this reason, Action Steps that a person is ultimately responsible for should remain on your list until completedeven when a person have delegated them to others. Simply marking that the Action Step has been delegated and to whom is sufficient.

Managerial Action Steps should be treated differently.

Aside from the Action Steps that are personal, there are three other types of Action Steps one should keep in mind as the leader of a project. The first type is delegated Action Steps. The second type is "Ensure Action Steps." Sometimes one will want to create an Action Step to ensure that something is completed properly in the future. Rather than being a nag to a team, one can create an Action Step that starts with the word "ensure." Creating "Ensure Action Steps" is a better alternative then sending numerous reminder emails to team members.

The last type of managerial Action Step is the "Awaiting" Action Step.

When one leaves a voicemail for someone, send a message to a potential customer, or respond to an email and clear it from the inbox, it is possible to forget to follow-up if the person fails to respond. So one should create an Action Step that starts with "Awaiting". In the online task manager one will set a target date for one week later. After a week passes, one will be reminded to follow up.

Foster an action-oriented culture.

A team needs an action-oriented culture to capitalise on creativity and effectiveness. It may feel a bit aggressive to ask people to capture an Action Step on paper, but fostering a culture in which such reminders are welcome helps ensure that Action Steps are not lost.

2. Team-motivation strategies and personal productivity

In the study, conducted by Dean Spitzer, as many as 50 percent of workers said they only put enough effort into their work to hold onto their jobs. And 84 percent said they could work better - if they wanted to. [1]

Employee motivation is usually treated as a problem of individual worker. Motivation programs and initiatives try to inspire employees to work harder, but they do nothing about the work conditions that continue to demotivate those same employees.

Thus motivating a team is often more challenging than motivating a single individual. Individuals within teams operate with different goals, values, beliefs, and expectations. Yet the variety of team member personalities can be a positive force if each performer contributes his or her unique capabilities when and where needed [13] in the correct environment.

2.1 Motivation in Daniel Pink's "Drive"

Daniel Pink in his book, called "Drive" shows that much of beliefs about motivation are not true. He also states the problem of a large number of organisations haven't caught up to the new understanding (introduced by Harlow and Deci a few decades ago), and operate from assumptions about human potential and individual performance that are outdated, unexamined, and rooted more in folklore than in science.

Systems of motivation

Author suggests to call a system of motivation, that is based on survival needs "Motivation 1.0". "Motivation 2.0" was built around external rewards and punishments, so called "carrots and sticks" method., but it's incompatible with some of the modern thinking, organisations and tasks. The next motivation system is called "Motivation 3.0."

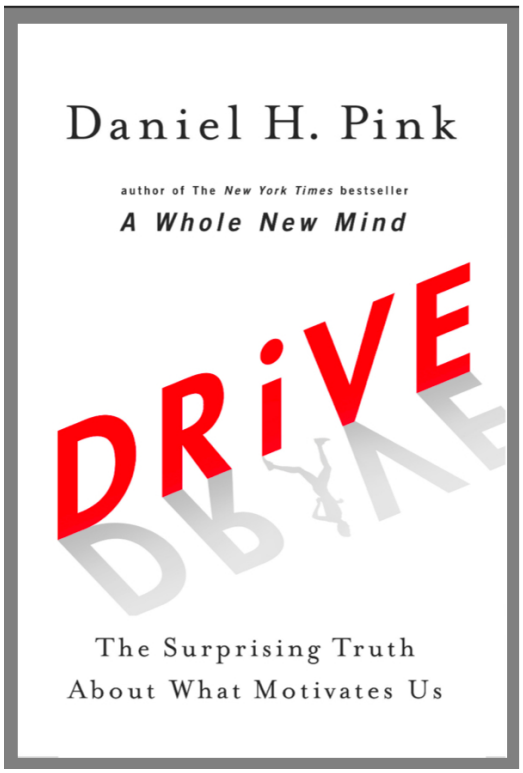

Figure 2.1: Daniel H. Pink, Drive

Experiment on inner motivation

The problem, existing within "Motivation 2.0" is greatly described by one of Lepper and Greenes early studies (which they carried out with a third colleague, Robert Nisbett) and has become a classic in the field and among the most cited articles in the motivation literature. The three researchers watched a classroom of preschoolers for several days and identified the children who chose to spend their "free play" time drawing. Then they fashioned an experiment to test the effect of rewarding an activity these children clearly enjoyed.

The researchers divided the children into three groups. The first was the "expected-award" group. They showed each of these children a "Good Player" certificateadorned with a blue ribbon and featuring the childs name – and asked if the child wanted to draw in order to receive the award. The second group was the "unexpected-award" group. Researchers asked these children simply if they wanted to draw. If they decided to, when the session ended, the researchers handed each child one of the "Good Player" certificates. The third group was the "no-award" group. Researchers asked these children if they wanted to draw, but neither promised them a certificate at the beginning nor gave them one at the end.

Two weeks later, back in the classroom, teachers set out paper and markers during the preschools free play period while the researchers secretly observed the students. Children previously in the "unexpected-award" and "no-award" groups drew just as much, and with the same relish, as they had before the experiment. But children in the first group – the ones whod expected and then received an award – showed much less interest and spent much less time drawing. The Sawyer Effect had taken hold. Even two weeks later, those alluring prizes – so common in classrooms and cubicles – had turned play into work.

To be clear, it wasnt necessarily the rewards themselves that dampened the

childrens interest. Remember: When children didnt expect a reward, receiving one had little impact on their intrinsic motivation. Only contingent rewards – if you do this, then youll get that – had the negative effect. Why? "If-then" rewards require people to forfeit some of their autonomy. Like the gentlemen driving carriages for money instead of fun, theyre no longer fully controlling their lives. And that can spring a hole in the bottom of their motivational bucket, draining an activity of its enjoyment.

Lepper and Greene replicated these results in several subsequent experiments with children. As time went on, other researchers found similar results with adults. Over and over again, they discovered that extrinsic rewardsin particular, contingent, expected, "if-then" rewardssnuffed out the third drive.

These insights proved so controversial – after all, they called into question a standard practice of most companies and schools – that in 1999 Deci and two colleagues reanalysed nearly three decades of studies on the subject to confirm the findings. "Careful consideration of reward effects reported in 128 experiments lead to the conclusion that tangible rewards tend to have a substantially negative effect on intrinsic motivation," they determined. "When institutionsfamilies, schools, businesses, and athletic teams, for examplefocus on the short-term and opt for controlling peoples behaviour," they do considerable long-term damage.

As one leading behavioural science textbook puts it, "People use rewards expecting to gain the benefit of increasing another persons motivation and behaviour, but in so doing, they often incur the unintentional and hidden cost of undermining that persons intrinsic motivation toward the activity."

This is one of the most robust findings in social science – and also one of the most ignored. Despite the work of a few skilled and passionate popularisers – in particular, Alfie Kohn, whose prescient 1993 book, Punished by Rewards, lays out a devastating indictment of extrinsic incentives – we persist in trying to motivate people this way.

The negative of Motivation 2.0

When carrots and sticks encounter ones third drive, strange things begin to happen. Traditional "if-then" rewards can give less of what a person want: They can extinguish intrinsic motivation, diminish performance, crush creativity, and

crowd out good behaviour. They can also give more of what one doesnt want: They can encourage unethical behaviour, create addictions, and foster short-term thinking (meaning not taking into account long-term outcome).

The positive of Motivation 2.0

Carrots and sticks are still can be an effective instrument for rule-based routine tasks – because theres little intrinsic motivation and not much creativity involved. And they can be more effective still if those giving such rewards offer a rationale for why the task is necessary, acknowledge that its boring, and allow people autonomy over how they complete it. For non-routine conceptual tasks, rewards are more perilous – particularly those of the "if-then" variety. But "now that" rewards – noncontingent rewards given after a task is complete – can sometimes be okay for more creative, right-brain work, especially if they provide useful information about performance.

Personality types and motivation

Daniel Pink also states, that Motivation 2.0 depended on and fostered Type X behaviour – behaviour fueled more by extrinsic desires than intrinsic ones and concerned less with the inherent satisfaction of an activity and more with the external rewards to which an activity leads. Motivation 3.0, the upgrade thats necessary for the smooth functioning of twenty-first-century business, depends on and fosters Type I behaviour. Type I behaviour concerns itself less with the external rewards an activity brings and more with the inherent satisfaction of the activity itself. For professional success and personal fulfilment, one need to move himself from Type X to Type I. Positevly, Type I people are made, not born – and Type I behaviour leads to stronger performance, greater health, and higher overall well-being.

Motivation

A person naturally seeks to be autonomous and self-directed. Unfortunately, circumstances – including outdated notions of "management" – often conspire to change that default setting and turn a person from Type I to Type X. To encourage Type I behaviour, and the high performance it enables, the

first requirement is autonomy. People need autonomy over task (what they do), time (when they do it), team (who they do it with), and technique (how they do it). Companies that offer autonomy, sometimes in radical doses, are outperforming their competitors.

Mastery

While Motivation 2.0 required compliance, Motivation 3.0 demands engagement. Only engagement can produce mastery –becoming better at something that matters. And the pursuit of mastery, an important but often dormant part of the third drive, has become essential to making ones way in the economy. Mastery begins with "flow"-optimal experiences (to be considered in detail in section 2.4) when the challenges faced are exquisitely matched to abilities. Smart workplaces therefore supplement day-to-day activities with "Goldilocks tasks" – not too hard and not too easy. But mastery also abides by three peculiar rules. Mastery is a mindset: It requires the capacity to see abilities not as finite, but as infinitely improvable. Mastery is a pain: It demands effort, grit, and deliberate practice. And mastery is an asymptote: Its impossible to fully realise, which makes it simultaneously frustrating and alluring.

Purpose

Humans, by their nature, seek purpose – a cause greater and more enduring than themselves. But traditional businesses have long considered purpose ornamental – a perfectly nice accessory, so long as it didnt get in the way of the important things. But thats changing – thanks in part to the rising tide of aging baby boomers reckoning with their own mortality. In Motivation 3.0, purpose maximisation is taking its place alongside profit maximisation as an aspiration and a guiding principle. Within organisations, this new "purpose motive" is expressing itself in three ways: in goals that use profit to reach purpose; in words that emphasize more than self-interest; and in policies that allow people to pursue purpose on their own terms. This move to accompany profit maximisation with purpose maximisation has the potential to rejuvenate our businesses and remake our world.

Goals

So how to deal with steps (goals) definition. Of course, all goals are not created equal. Goals and extrinsic rewards arent inherently corrupting. But goals are more toxic than Motivation 2.0 recognises. In fact, the business school professors suggest they should come with their own warning label: Goals may cause systematic problems for organisations due to narrowed focus, unethical behaviour, increased risk taking, decreased cooperation, and decreased intrinsic motivation. One should use care when applying goals in organisation.

- Offering a rationale for why the task is necessary. A job thats not inherently interesting can become more meaningful, and therefore more engaging, if its part of a larger purpose. For example: "Explain why this poster is so important and why sending it out now is critical to your organisations mission."
- Acknowledge that the task is boring. This is an act of empathy, of course. And the acknowledgment will help people understand why this is the rare instance when "if-then" rewards are part of how organisation operates.
- Allow people to complete the task their own way. Think autonomy, not control. State the outcome you need. But instead of specifying precisely the way to reach ithow each poster must be rolled and how each mailing label must be affixedgive them freedom over how they do the job.

Do's and dont's

- Offering an "if-then" reward to the design staff. A one shouldn't stride into the office and announce: "If you come up with a poster that rocks my world or that boosts attendance over last year, then youll get a ten-percent bonus." Although that motivational approach is common in organisations all over the world, its a recipe for reduced performance. Creating a poster isnt routine. It requires conceptual, breakthrough, artistic thinking. The best approach is to have already established the conditions of a genuinely motivating environment. The baseline rewards must be sufficient. That is, the teams basic compensation must be adequate and fair –particularly compared with people doing
- Similar work for similar organisations. A nonprofit must be a congenial place to work. And the people on team must have autonomy, they must have ample opportunity to pursue mastery, and their daily duties must

relate to a larger purpose. If these elements are in place, the best strategy is to provide a sense of urgency and significanceand then get out of the talents way. But one may still be able to boost performance a bit – more for future tasks than for this one – through the delicate use of rewards. Efforts will backfire unless the rewards offered meet one essential requirement.

- The essential requirement: Any extrinsic reward should be unexpected and offered only after the task is complete. Holding out a prize at the beginning of a project – and offering it as a contingency – will inevitably focus peoples attention on obtaining the reward rather than on attacking the problem. But introducing the subject of rewards after the job is done is less risky. In other words, where "if-then" rewards are a mistake, shift to "now that" rewards – as in "Now that youve finished the poster and it turned out so well, Id like to celebrate by taking you out to lunch." As Deci and his colleagues explain, "If tangible rewards are given unexpectedly to people after they have finished a task, the rewards are less likely to be experienced as the reason for doing the task and are thus less likely to be detrimental to intrinsic motivation." So when the poster turns out great, one could buy the design team a case of beer or even hand them a cash bonus without snuffing their creativity. The team didnt expect any extras and getting them didnt hinge on a particular outcome. A person simply offers his appreciation for their stellar work. But repeated "now that" bonuses can quickly become expected "if-then" entitlementswhich can ultimately crater effective performance. A firmer motivational footing is created if follow two additional principles.

  Consider nontangible rewards. Praise and positive feedback are much less corrosive than cash and trophies. In fact, in Decis original experiments, and in his subsequent analysis of other studies, he found that "positive feedback can have an enhancing effect on intrinsic motivation."

  Provide useful information. Amabile has found that while controlling extrinsic motivators can clobber creativity, "informational or enabling motivators can be conducive" to it. In the workplace, people are thirsting to learn about how theyre doing, but only if the information isnt a tacit effort to manipulate their behaviour. The following shouldn't be permitted:

"That poster was perfect. You did it exactly the way I asked." Instead, people should be given a meaningful information about their work. The more feedback focuses on specifics ("great use of colour") – and the more the praise is about effort and strategy rather than about achieving a particular outcome – the more effective it can be.

2.2 Goal commitment formula by Richard Clark

While Daniel Pink provides a distinction between different types of tasks and motivation behind them, Richard Clark did a great study on the components of goal commitment.

Cark's paper quotes Hogan Curphy and Hogan review of leadership studies: found that only about 30 percent of line managers are able to adequately motivate the people who report to them. They imply that in most circumstances, motivation accounts for about half of all performance results.

The late Tom Gilbert, one of the clearest thinkers in performance improvement, was fond of saying that when two people had equal abilities, the enthusiastic member of the pair would achieve about 70 percent more than the unenthusiastic person. Even more troubling is that evidence that a majority of the published studies of organisational development strategies that report measured increases in motivation are fatally flawed. Strategies that may not work as powerfully or as consistently as claimed include popular employee empowerment strategies, contests, job redesign, leaderless teams and various performance recognition techniques.

In the CANE model, motivation is defined as two interlinked processes. The first process leads us to make a commitment to a performance goal and persist the face of distractions from appealing but less important alternative goals. The second motivation process is concerned with the amount and quality of the mental effort people invest in achieving the knowledge component of performance goals. These two motivation processes, committed, active and sustained goal pursuit on the one hand, and necessary mental effort to tackle goal-related problems, on the other hand, are the primary motivation goals in the CANE model.

In todays complex work environments the variety of job tasks that confront

KNOWLEDGE Makes it possible to:	**MOTIVATION** Makes it possible to:
Understand goals	Make a strong commitment to goals
Develop/implement plans to reach goals	Persist with plans and goals when distracted
Monitor progress towards goals	Monitor goal value and personal confidence
Revise and improve plans	Invest necessary mental effort in plans

Figure 2.2: Knowledge and motivation systems

all of us change constantly over time. We cannot commit ourselves equally to all tasks. We must prioritise and focus on important tasks in order to be successful. Commitment problems happen when people resist assigning adequate priority to important job tasks. Research on motivation suggests that people with commitment problems may avoid a task altogether and/or argue that the task is less important than some other set of tasks.

Three factors have been found to increase (or decrease) work goal commitment. The first factor is **"task assessment".** All of us will analyse any task we are assigned to determine whether we can successfully complete the task. We all tend to ask ourselves two questions about new tasks - "Can I do it?" and Will I be permitted to do it?. If we think that we have the ability to accomplish the goal and that we will be permitted to accomplish it, our commitment will increase. If we doubt our ability or the organisations willingness to let us use our skills, commitment will decrease.

Emotion and commitment. The second factor influencing commitment is our mood or emotions. All positive emotions facilitate commitment and all powerful, negative emotions discourage goal commitment. This may seem like a minor issue but for temperamental people or in organisations where pressure is high and/or change is constant, negative emotional undercurrents can be strong. Angry or depressed people find it nearly impossible to make a commitment to work goals.

Values and commitment. The final factor that influences the strength

TASK ASSESSMENT	×	**EMOTION**	×	**VALUES**	=	**GOAL COMMITMENT**
Ability (Can I do it?) Context (Will I be permitted?)		(Do I feel like it?)		Utility (Value later?) Interest (Am I curious?) Importance (Is this me?) *Interdependence (Does anybody count on me?)*		

Figure 2.3: CANE Model of Factors Influencing Goal Commitment

of goal commitment is our personal value in the goal. It is my experience that values are the most important element in increasing or decreasing the strength of our commitments. Psychologists now have good evidence that the most important value at work is our belief about whether the achievement of a work goal will increase our personal control or effectiveness (Shapiro et al, 1996; Locke and Latham, 1990). The more we believe that achievement of a work goal will make us more successful, the higher our level of commitment to the goal. The reverse is also true. Few of us will give a high priority to tasks that we sincerely believe will lead us to fail or be perceived as incompetent Utility, Interest and Importance Values.

Task Assessment Solutions

Solving task assessment problems require that we convince people that they can do a job and that existing barriers to their performance will be removed. Pointing out familiar, past examples of job performance that are similar to the new task helps increase confidence. In addition, job aids can bolster confidence. Involving staff in the elimination of any procedural or policy barriers to performance reduces resistance based on task assessment. The service technicians had excellent job aids which increased their confidence about the form task. The key element here is to persuade or empower people to believe that they can succeed at the task they are avoiding. Bandura provides extensive examples of solutions in this area.

Mood solutions.

Mood problems often take more time to develop than task assessment or value problems. I find mood problems to be key elements in organisations where a major culture or job change is occurring. This is particularly true in organisations that are changing from a civil service to a business culture.

Solutions that have been found to change mood states have included listening to positive mood music; writing or telling about a positive mood-related experience; watching a movie or listening to stories that emphasize positive mood states; and emotion control training through "environmental control strategies" including the choice of how we complete work tasks, adjusting work space and "positive self talk".

Value solutions

The solution to most commitment problems and opportunities is to convince people that completing the task they are resisting will make them more effective and/or perceived as more effective. People simply will not do what they believe will make them less effective or less successful. Many people are suspicious of change simply because they feel that they will be perceived as less effective under novel, negative or uncertain conditions. They must be convinced that if they commit themselves to the avoided task(s) they will become significantly more effective or successful. The specific solution that accomplishes this goal may be quite different for different individuals and work cultures. Some organisations have adopted various "employee empowerment" solutions to value problems. In many empowerment settings staff are asked to choose their own work goals in order to get them to value their work. There is good research evidence that this is not necessary.

In cases where participatory goal setting is not possible, they find that value for the goal is enhanced if people perceive the goal to be: 1) assigned by a legitimate, trusted authority with an "inspiring vision" that reflects a "convincing rationale" for the goal (importance value), and who; 2) provides expectation of outstanding performance (importance value) and gives: 3) "ownership" to individuals and teams for specific tasks (interest value); 4) expresses confidence in individual and team capabilities (interest value) while; 5) providing feedback on progress that includes recognition for success and supportive but corrective suggestions for mistakes (utility value).

Motivation Solutions

Value problems are often multi-level issues in an organisation. In this organisation, there were a number of beliefs and patterns that had to be considered. The mangers of the technicians had their own motivational issues to handle. For example, the senior manager acting as sponsor for the motivation project placed a number of constraints on a value solution for the technicians.

Two types of motivation are important at work, persistence and mental effort. Commitment (persistence at a task) is increased by convincing people that: a) the organisation will remove unnecessary barriers; b) that achievement

of the work goal will make the person more personally effective; and c) that the manager requesting the goal is credible, trustworthy, optimistic (about the person or teams ability to achieve the goal), able to clearly communicate the vision connected to the goal and willing to give ownership for the accomplishment. Mental effort is enhanced by insuring that the goal assigned is very challenging. Managers must work with people to adjust their confidence level whenever they become over confident (and thus refuse to take responsibility for errors or poor performance) or under confident (and thus find an excuse to procrastinate or avoid the goal altogether).

2.3 Gamification

Following the success of the location-based service Foursquare, the idea of using game design elements in non-game contexts to motivate and increase user activity and retention has rapidly gained traction in interaction design and digital marketing. Under the moniker "gamification", this idea is spawning an intense public debate as well as numerous applications ranging across productivity, finance, health, education, sustainability, as well as news and entertainment media. Several vendors now offer "gamification" as a software service layer of reward and reputation systems with points, badges, levels and leader boards.

This commercial deployment of 'gamified' applications to large audiences potentially promises new, interesting lines of inquiry and data sources for human-computer interaction (HCI) and game studies and indeed, "gamification" is increasingly catching the attention of researchers.

Whereas "serious game" describes the design of full-fledged games for non-entertainment purposes, "gamified" applications merely incorporate elements of games (or game "atoms" [10]). Of course, the boundary between "game" and "artifact with game elements" can often be blurry is Foursquare a game or a "gamified" application? To complicate matters, this boundary is empirical, subjective and social: Whether you and your friends 'play' or 'use' Foursquare depends on your (negotiated) focus, perceptions and enactments. The addition of one informal rule or shared goal by a group of users may turn a 'merely' 'gamified' application into a 'full' game. Within game studies, there is an

increasing acknowledgement that any definition of 'games' has to go beyond properties of the game artifact to include these situated, socially constructed meanings. For the present purpose, this means that (a) artifactual as well as social elements of games need to be considered, and (b) artifactual elements should be conceived more in terms of affording gameful interpretations and enactments, rather than being gameful. Indeed, the characteristic of 'gamified' applications might be that compared to games, they afford a more fragile, unstable 'flicker' of experiences and enactments between playful, gameful, and other, more instrumental-functionalist modes.

As can be seen, this level model distinguishes interface design patterns from game design patterns or game mechanics. Although they relate to the shared concept of pattern languages, unlike interface design patterns, neither game mechanics nor game design patterns refer to (prototypical) implemented solutions; both can be implemented with many different interface elements. Therefore, they are more abstract and thus treated as distinct.

So to restate, whereas serious games fulfill all necessary and sufficient conditions for being a game, gamified applications merely use several design elements from games. Seen from the perspective of the designer, what distinguishes gamification from regular entertainment games and serious games is that they are built with the intention of a system that includes elements from games, not a full game proper. From the user perspective, such systems entailing design elements from games can then be enacted and experienced as games proper, gameful, playful, or otherwise this instability or openness is what sets them apart from games proper for users.

Similar to serious games, "gamification" uses elements of games for purposes other than their normal expected use as part of an entertainment game. Now 'normal use' is a socially, historically and culturally contingent category. However, it is reasonable to assume that entertainment currently constitutes the prevalent expected use of games. Likewise, joy of use, engagement, or more generally speaking, improvement of the user experience represent the currently predominant use cases of "gamification".

To summarize: "Gamification" refers to

- the use (rather than the extension) of
- design (rather than game-based technology or other game- related prac-

tices)

- elements (rather than full-fledged games)
- characteristic for games (rather than play or playfulness)
- in non-game contexts (regardless of specific usage intentions, contexts, or media of implementation).

This definition contrasts "gamification" against other related concepts via the two dimensions of playing / gaming and parts / whole. Both games and serious games can be differentiated from "gamification" through the parts/whole dimension. Playful design and toys can be differentiated through the playing/gaming dimension.

2.3.1 Appropriate game mechanics for the basic project management

The following list of requirements for successful productivity application was conducted (the full list of game mechanics can be found in appendix A):

- "Achievement" game mechanics supports desire to perform tasks
- "Pride" game mechanics supports retention of joy from executed tasks
- "Avoidance" game mechanics helps a person return to application
- "Cascading Information Theory" allows to introduce difficult concepts of GTD (Getting Things Done) and increase a persons activity in the application
- "Communal Discovery" is used in collaborative tasks / task delegation
- "Progression Dynamic" helps visualize improvement and progress to a goal.

2.4 Mihaly Csikszentmihalyi's concept of "Flow"

The author has been studying for over 20 years the states of optimal experience – those times when people report feelings of concentration and deep enjoyment. These investigations have revealed that what makes experience genuinely satisfying is a state of consciousness called flow–a state of concentration so focused that it amounts to absolute absorption in an activity.

Everyone experiences flow from time to time and will recognise its characteristics: people typically feel strong, alert, in effortless control, unselfconscious, and at the peak of their abilities. Both a sense of time and emotional problems

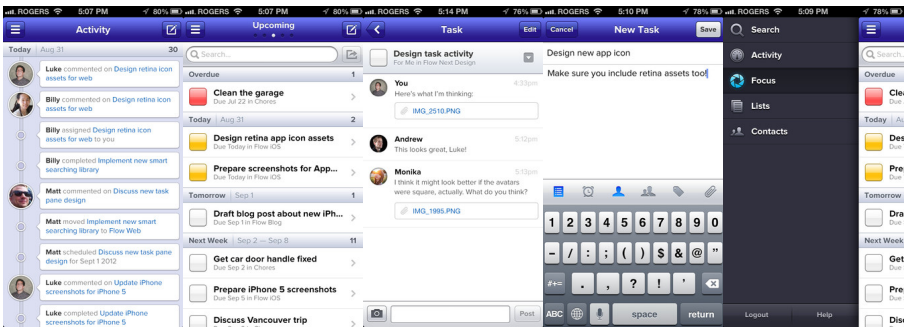

Figure 2.4: Challenge to Skill Relation

seem to disappear, and there is an exhilarating feeling of transcendence. Flow: The Psychology of Optimal Experience describes how this pleasurable state can be controlled, and not just left to chance, by setting ourselves challenges – tasks that are neither too difficult nor too simple for our abilities. With such goals, we learn to order the information that enters consciousness and thereby improve the quality of our lives.

The studies have suggested that the phenomenology of enjoyment has eight major components. When people reflect on how it feels when their experience is most positive, they mention at least one, and often all, of the following:

- We confront tasks we have a chance of completing;
- We must be able to concentrate on what we are doing;
- The task has clear goals;
- The task provides immediate feedback;
- One acts with deep, but effortless involvement, that removes from awareness the worries and frustrations of everyday life;
- One exercises a sense of control over their actions;
- Concern for the self disappears, yet, paradoxically the sense of self emerges stronger after the flow experience is over; and
- The sense of duration of time is altered. The combination of all these elements causes a sense of deep enjoyment that is so rewarding people feel that expending a great deal of energy is worthwhile simply to be able to feel it.

A Challenging Activity that Requires Skills

Optimal experiences are reported to occur within sequences of activities that are goal-directed and bounded by rules–activities that require the investment of psychic energy (attention) and that could not be done without skills.

Please note that activities do not need to be physical and skills also need not be physical skills. For instance, the most frequently mentioned enjoyable activity the world over was reading, followed closely by being with other people. For those who do not have the right skills, an activity is not challenging; it is simply meaningless. Challenges of competition were found to be stimulating and enjoyable. But when beating the opponent takes precedence in the mind over performing as well as possible, enjoyment tends to disappear. Competition is enjoyable only when it is a means to perfect one's skills; when it becomes an end in itself, it ceases to be fun.

The Merging of Action and Awareness

One of the most universal and distinctive features of optimal experience is the people become so involved in what they are doing that the activity becomes spontaneous, almost automatic; they stop being aware of themselves as separate from the actions they are performing. It often requires strenous physical exertion, or highly disciplined mental activity to enter a continuous flow.

Clear Goals and Feedback

Unless a person learns to set goals and to recognize and gauge feedback in their activities, she will not enjoy them. For activities that are creative or open-ended in nature, a person must develop a strong sense of what she intends to do or negotiate goals and rules during the activity. These goals and rules provide benchmarks for feedback. The kind of feedback we work toward is in, and of itself, often unimportant. What makes feedback valuable is the symbolic message it contains: that I have succeeded in my goal.

Concentration on the Task at Hand

One of the most frequently mentioned dimensions of the flow experience is that, while it lasts, one is able to forget all the unpleasant aspects of life. The task requires such concentration that only a very select range of information can be allowed into awareness.

The Paradox of Control

The flow experience is typically described as involving a sense of control–or more precisely, as lacking the sense of worry about losing control that is typical in many situations of normal life. What people enjoy is not the sense of being in control, but the sense of exercising control in difficult situations. However, when a person becomes dependent on the ability to control an enjoyable activ-

ity then he loses the ultimate control: the freedom to determine the content of consciousness. While experiences are capable of improving the quality of existence by creating order in the mind, they can also become addictive, at which point the self becomes captive of a certain kind of order, and is then unwilling to cope with the ambiguities of life.

The Loss of Self-Consciousness

When in a flow experience, what slips below the threshold of awareness is the concept of self, the information we use to represent to ourselves who we are. And being able to forget temporarily who we are seems to be very enjoyable. When not preoccupied with our selves, we actually have a chance to expand the concept of who we are. Loss of self-consciousness can lead to self-transcendence, to a feeling that the boundaries of our being have been pushed forward.

The Transformation of Time

One of the most common descriptions of optimal experience is that time no longer seems to pass the way it ordinarily does. Generally, after the experience we do not know where the time went; however, during the actual experience, time seems to stand still.

The key element of an optimal experience is that it is an end in itself. It is an autotelic experience. The term "autotelic" derives from two Greek words, "auto" meaning self, and "telos" meaning goal. It refers to a self-contained activity, one that is done not with the expectation of some future benefit, but simply because the doing itself is the reward. Teaching children in order to turn them into good citizens is not autotelic, whereas teaching them because one enjoys interacting with children is. Most enjoyable activities are not natural; they demand an effort that initially one is reluctant to make. But once the interaction starts to provide feedback to the person's skills, it usually begins to be intrinsically rewarding.

Flow in the family context has five characteristics:

- **Clarity:** children know what parents expect from them;
- **Centering:** children know that their parents are interested in what they are doing in the present;
- **Choice:** children feel that they have a variety of possibilities from which to choose;
- **Commitment:** trust that allows the child to feel comfortable enough to

set aside the shield of defenses and become unself-consciously involved; and

- **Challenge:** providing increasingly complex opportunities for action.

2.5 General guidelines

These guidelines were combined with advices from "Winning without losing" by Martin Bjergegaard, Jordan Milne.

Personal guidelines:

1. A list of things, that take your time, but don't contribute neither to effectiveness nor to happiness is effective
2. It is OK to force yourself to do things that engage
3. A person should value and appreciate things you did today
4. Often visit the open air
5. Know your plan B (maybe secondary tasks for today)
6. Split the day between different spheres of life, 8-8-8
7. Happiness is to stay active and not to be tired in taking new tasks, which help you develop and learn
8. Don't create a long business-plans
9. See the big picture
10. Don't micromanage, let others do their work
11. Don't try to get everything done, just the important
12. Lucid respites are important
13. A list for today (fixed length)
14. Lean
15. Not a schedule, but inspirationz
16. It is better to solve bad tasks at first

3. Productivity mobile applications analysis

Let's consider the most popular productivity mobile applications on the iOS devices in order to understand market and how does these application solve the problem of effective task management.

3.1 Popular applications and their description

Habit List. http://habitlist.com/ Habit List is an application to create good habits and break unhealthy ones. Habit List allows users to track habits and provide visual representations on progress. This encourages more success by seeing the results of hard work or lack thereof. The app is quick to load and the design is simple but effective. Its essentially a list of items that a user would like to complete on a certain schedule. Each habits status is communicated through a green or red indicator. Inside the colored indicator is the streak of days that a habit has been completed or not been completed.

Carrot. http://meetcarrot.com/ Carrot encourages users toget tasks done in a timely matter and they're rewarded with points that lead to unlockable features. But if they don't accomplish tasks then the app will get angry.

Clear. http://www.realmacsoftware.com/clear/ Clear uses the hierarchy of color, which is utilized to represent priority within the app. The idea is that to-do list becomes a heat map, where the warmest areas (i.e.

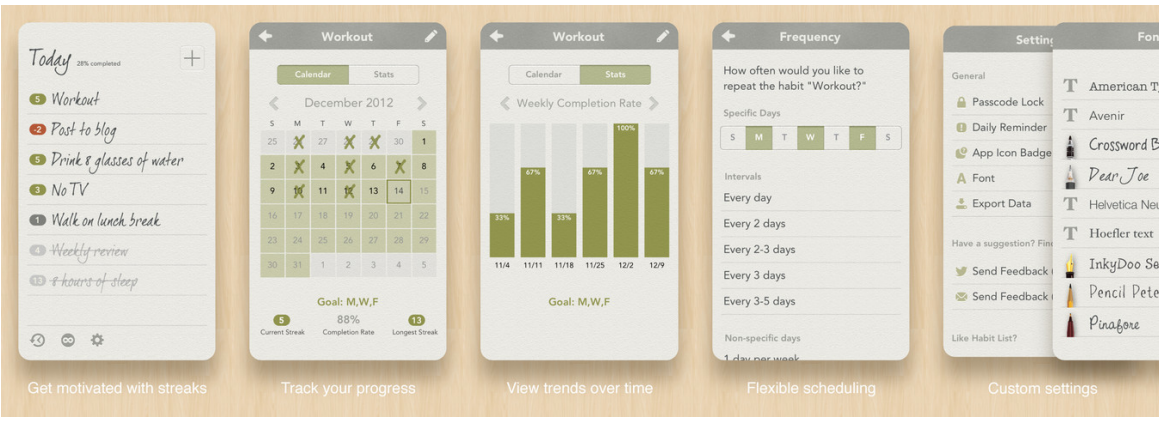

Figure 3.1: Habit list

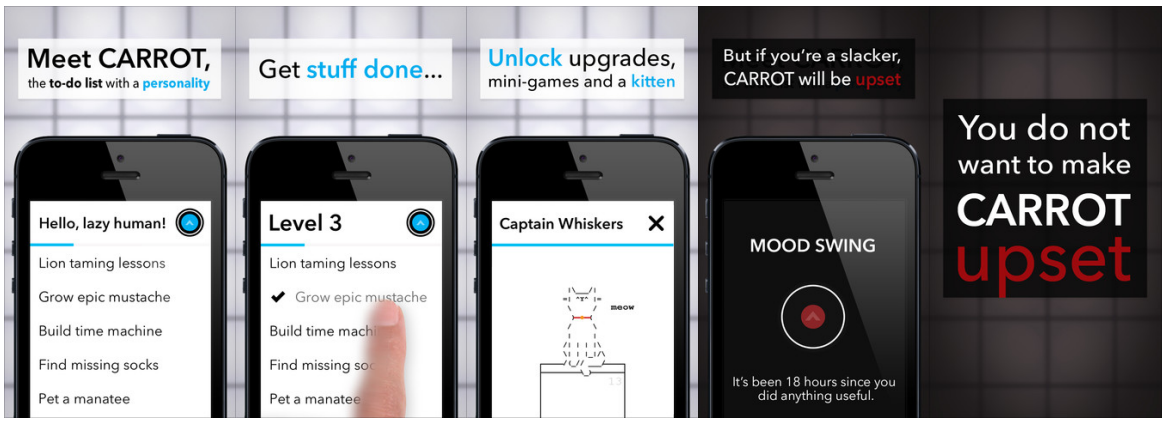

Figure 3.2: Carrot

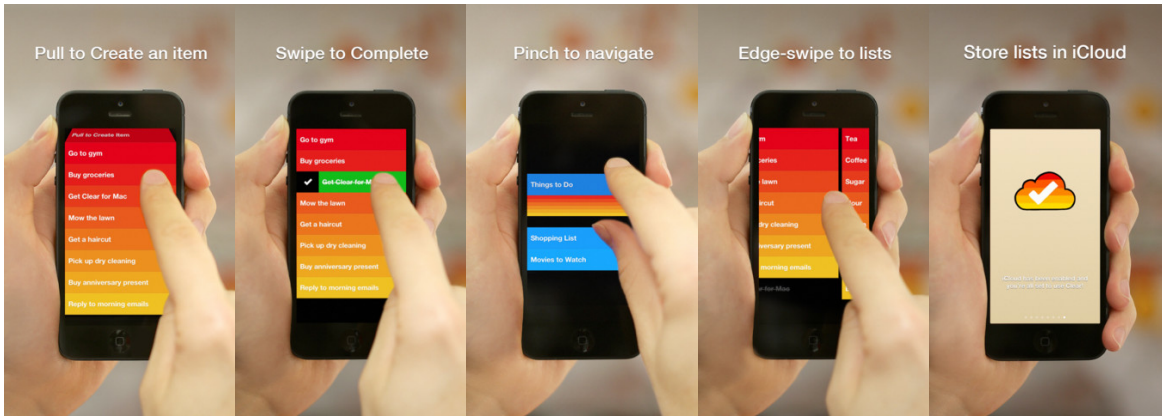

Figure 3.3: Clear

the ones with the strongest shade of red) are the tasks should be attended first.

Remember the milk. http://www.rememberthemilk.com/ Remember the milk offers a lot of GTD functionality: tag clouds, custom task lists, geo location, Gmail extension, Google Calendar integration, etc.

Wunderlist 2. https://www.wunderlist.com Wunderlist 2 comes with a number of features: online syncing, filtered lists, and Facebook integration, collaboration. The design spans across the multitude of different devices Wunderlist works on, giving it a single, unified interface.

Flow. http://www.getflow.com/ Flow provides all the collaboration tools needed to manage a project (files, deadlines, tasks and discussion) online in one centralized place for everyone. It allows to discuss projects, set deadlines, take notes, and keep everyone on the same page online, even when they're out of the office.

Things. http://culturedcode.com/things/iphone/ Things features a daily review to plan a day on the go: to-dos that are scheduled for today

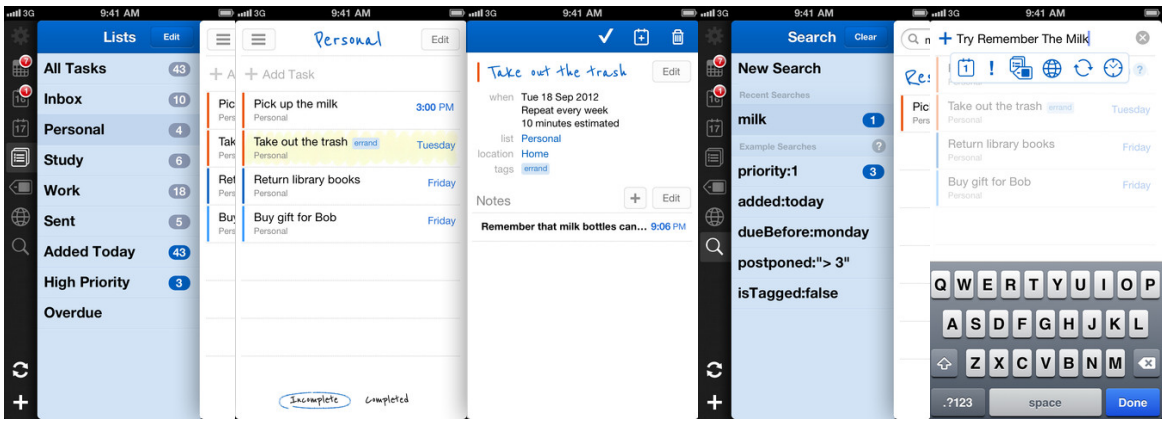

Figure 3.4: Remember the milk

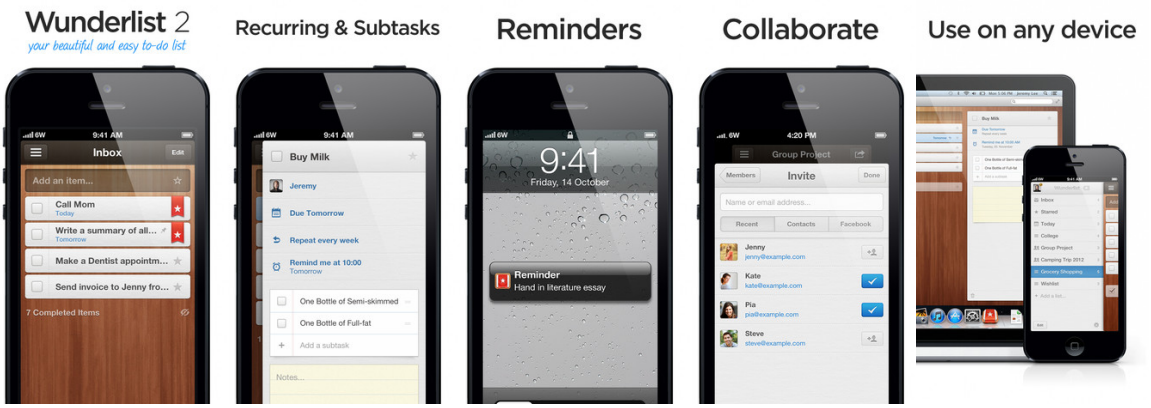

Figure 3.5: Wunderlist

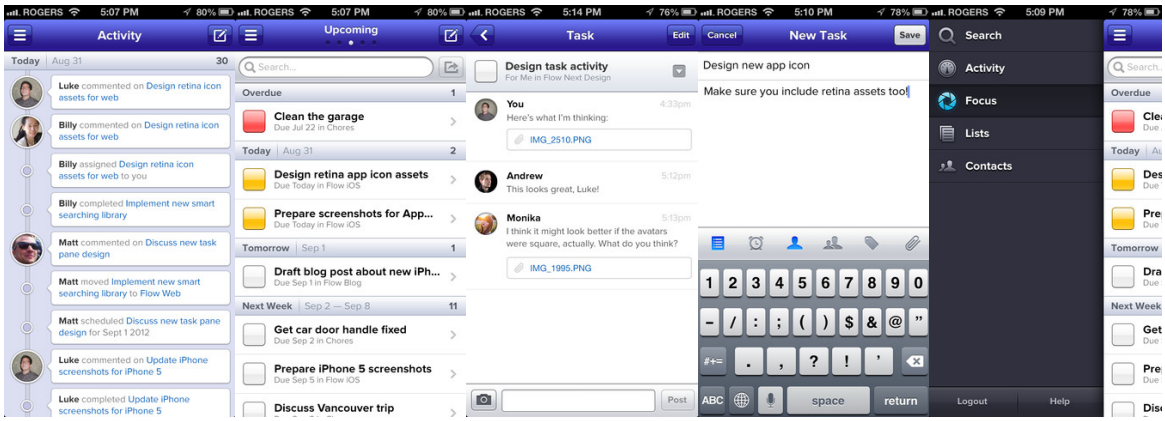

Figure 3.6: Flow

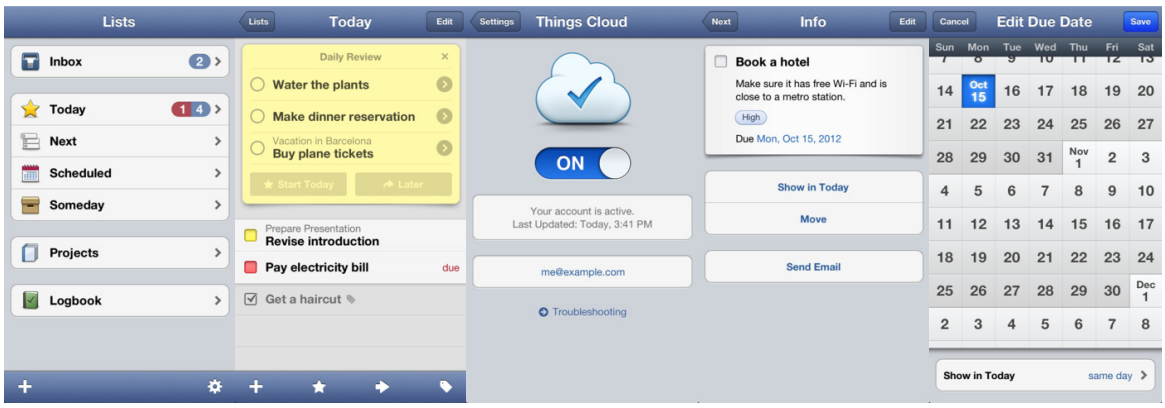

Figure 3.7: Things

automatically appear at the top of Today list, together with to-dos that have become due; instant action to find the right tasks for the current context using tags; predefined lists, such as “today”, “next”, “scheduled”, “someday”.

3.2 Game mechanics in use

Most of the listed applications made a different approach to these mechanics. But only one application tried to develop an emotional connection to a person. It is Carrot, which uses natural language and colour-coded emotions in order to response to actions of a person. If a person didnt achieve anything in a period of time, Carrot becomes angry. In addition, this application used the most of introduced game mechanics (see table 3.1).

Achievement definition: A virtual or physical representation of having accomplished something. These are often viewed as rewards in and of themselves. Pride definition: the feeling of ownership and joy at an accomplishment.

Table 3.1: Popular applications and used game mechanics

Application	**Game Mechanics**
Habit List	Achievement, pride, avoidance
Carrot	Cascading Information Theory, pride, avoidance, achievement, progression dynamic
Clear	Cascading Information Theory, Achievement
Remember the milk	Achievement

Table 3.1: Popular applications and used game mechanics

Application	**Game Mechanics**
Wunderlist 2	Achievement, pride
Flow	Achievement, communal discovery
Things	Achievement

Avoidance definition: the act of inducing player behavior not by giving a reward, but by not instituting a punishment. Produces consistent level of activity, timed around the schedule.

Communal Discovery definition: The game dynamic wherein an entire community is rallied to work together to solve a riddle, a problem or a challenge. Immensely viral and very fun. Cascading Information Theory definition: The theory that information should be released in the minimum possible snippets to gain the appropriate level of understanding at each point during a game narrative.

Progression dynamic definition: this is a dynamic in which success is granularly displayed and measured through the process of completing itemised tasks.

3.3 What to learn from productivity apps

Summarizing the conducted results, the following list of requirements for successful productivity application was made:

- "Achievement" game mechanics supports desire to perform tasks
- "Pride" game mechanics supports retention of joy from executed tasks
- "Avoidance" game mechanics helps a person return to application
- "Cascading Information Theory" allows to introduce difficult concepts of GTD (Getting Things Done) and increase a persons activity in the application
- "Communal Discovery" is used in collaborative tasks / task delegation
- "Progression Dynamic" helps visualise improvement and progress to a goal.

Techniques used in animation to create believable characters can help to establish an emotional connection to the person.

4. Methodology for stimulating high-performance of project participants

4.1 Team-wide productivity

Motivating a team is often more challenging than motivating a single individual. Individuals within teams operate with different goals, values, beliefs, and expectations. Yet the variety of team member personalities can be a positive force if each performer contributes his or her unique capabilities when and where needed.

The first critical issue in team motivation is to be clear about the definition of "a team." Nearly everyone who studies teams emphasizes that it is unnecessary to use team motivation strategies when teams are defined as any group of two or more people with similar skills who are simply working together to achieve a common goal. For a team to exist (for motivational purposes), team members must play different roles or bring different skills to the table. Those different skills must be required to achieve team goals. So a team is an interdependent group of individuals, each possessing a different set of skills but who collectively possess all of the skills required to achieve team goals.

Foster Mutual Respect for the Expertise of All Members

Teams on which one or more members believe that they are working with people who lack adequate skills to achieve team goals have a major motivational problem. In some cases, this belief is simply incorrect. Highly competitive people sometimes distort the real situation and develop the self-protective view that one or more people on their team are inadequate. Competitive spirit is good. But bolstering self-confidence at the expense of others is immature and destructive. Bandura describes many studies in a variety of fields where "weak link" doubts about team member expertise have significantly reduced team effectiveness. Even though all team members vary in their expertise levels, when individuals respect and support one another, less-able team members tend to perform significantly better and work hard over time to increase their skills. Since individual team members tend to be self-focused and so think more

about their own contributions and ability, team members need to be reminded about the skills of other members. One effective way to accomplish this task is to actively attribute successes to each team members expertise.

Help Weaker Members Believe That Their Effort Is Vital to Team Success

Occasionally teams must accommodate members who are novices or who for some reason are not able to do the best job for the team. When teams cant replace weaker members, what works best to preserve team motivation? Jackson and LePine (2003) have recent and solid evidence that when team members believe that their weakest member is merely inexperienced or has faltered for some uncontrollable reason (for example, illness, accident, or a family crisis) and can improve, they will give support provided that the person is investing effort to do so.

The biggest motivational challenge on a team is faced by the weakest member. That individual must believe that what he or she contributes to the team is vital to the teams success and that the other members expect him or her to improve and succeed. Feedback to members who are working to improve must emphasize effort, not ability. When they make progress, it is best to attribute the progress to effort. When no progress is forthcoming, they need to be urged to "get busy, get serious and work harder." Avoid attributing success or failure to ability. Belief that performance is due to ability tends to discourage hard work.

In many teams the motivational challenge is not a weak link, but instead a lack of cooperation and collaboration.

Support a Shared Belief in the Teams Cooperative Capabilities

Healthy teams are made up of team players who cooperate with each other. One uncooperative person can damage the motivation of even the most capable team. The obvious example is the arrogant, self-focused prima donna who invests most of his or her effort trying to look good with managers and clientsat the expense of the team. Less obvious but equally destructive is the outwardly supportive but silently devious back-stabber, whose primarily goal is to make his or her own work highly visible.

Hold Individual Members Accountable for Contributions to the Team Effort

One of the first team motivation studies, performed just after the turn of the century, established the principle that has been called "social loafing." When people pulled as hard as possible against a rope connected to a strain gage, their best effort was recorded. When another person was added to the rope and two people pulled together, each person invested less effort in a collaborative effort than he or she did when alone. As more people were added to the rope, each person pulled less forcefully. When interviewed, most people seem unaware that they are not working as hard in a group situation as they did when alone.

Direct the Teams Competitive Spirit Outside the Team and the Organization

Competition can be highly motivating for individuals or teams. Salespeople seem to thrive on it, and many people who are raised in Western cultural traditions seem to like a bit of it. One of the most common motivational team-build- ing exercises favored by organizational consultants is a field experience where teams compete with other teams to bond and build team spirit. These events are scheduled off site and are ideally held in unfamiliar settings to interrupt habitual patterns formed at work for relating to others. Teams are challenged to do something highly novel, such as build structures or navigate difficult terrain to reach a tar- get sooner or more effectively than other teams. Individuals are asked to notice how hard they are working, how much they are collaborating, and whether they have a real desire to "win."

Teams are defined as collections of individuals with different skill sets working together to achieve goals that require members to collaboratively apply their different skills. Collections of individuals with similar skills who tackle problems do not require team motivation strategies. In addition to motivational strategies that work with individuals, interdependent teams are most motivated when they trust both the expertise and collaborativeness of other team members as well as the determination of weaker members on Direct the Teams Competitive Spirit Outside the Team and the Organisation Competition can be highly motivating for individuals or teams. Salespeople seem to thrive on it, and many people who are raised in Western cultural traditions seem to like a bit of it. One of the most common motivational team-building exercises favoured by organisational consultants is a field experience where teams compete with other teams to bond and build team spirit. These events are scheduled off site and

are ideally held in unfamiliar settings to interrupt habitual patterns formed at work for relating to others. Teams are challenged to do something highly novel, such as build structures or navigate difficult terrain to reach a tar- get sooner or more effectively than other teams. Individuals are asked to notice how hard they are working, how much they are collaborating, and whether they have a real desire to "win."

In general, team-building exercises have been found to be very effective, but they also have a potentially ugly, unintended side effect. Druckman and Bjork (1994) reviewed all studies of team building for the US National Academy of Sciences. The variety of team-building methods shared the common goal of attempting to get members of work teams to bond, collaborate, and work efficiently toward common goals by competing with other teams. The researchers concluded that many different approaches worked, but they were surprised to find that after team-building exercises, a significant number of teams were competing in a nearly suicidal fashion with other teams in their own organisation. Stories include misguided team members who were found to be modifying or deleting the electronic files, intentionally "misplacing" or rerouting team resources, and spreading negative rumors about members of other teams in their organizations. Apparently, fostering constant, intense rivalry can help when it is directed at the organizations competition, but it can also engender a destructive level of internal competition and focus attention and energy away from organizational goals. The obvious motivational issue in this situation is to make certain that team-building exercises focus the teams competitive energy on competing organizations – not on other teams within the same organization.

Summary

Teams are defined as collections of individuals with different skill sets working together to achieve goals that require members to collaboratively apply their different skills. Collections of individuals with similar skills who tackle problems do not require team motivation strategies. In addition to motivational strategies that work with individuals, interdependent teams are most motivated when they trust both the expertise and collaborativeness of other team members as well as the determination of weaker members on their team to invest maximum effort to build their expertise. In addition, team members must believe

that their own contributions to the team effort are being constantly and fairly evaluated along with the performance of the entire team. Finally, team competitiveness must be focused on opposing organizations that are struggling for the same customer base, not on teams in their own organization.

5. Building productivity mobile application

In this chapter the prototype of personal performance mobile application will be considered. Let's call it "Happy Perform" (happy-perform.com)

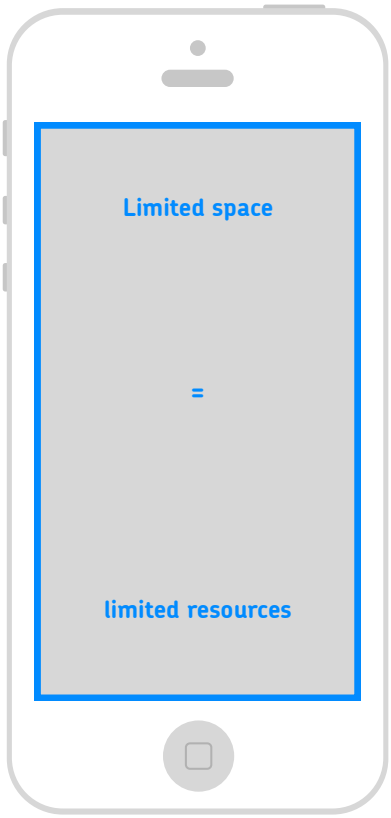

Figure 5.1: Happy Perform, limited space and resources

5.1 Abstractions and games

There is an important notion of feedback within the concept of "Flow" and appearing in all of the method of project management.

The only application from considered ones in chapter 3 did try to use a concept of feedback: "Carrot". It used a game mechanic called "Negative reinforcement" in order to state a bad system condition: there is a lot of tasks, that require attention.

5.1.1 Concept of time limitation

Each task (that has a deadline) can be represented as a system with dynamic complexity. As focus is a limited resource, as a time, tasks that don't get enough attention can lead to a bad state of a system very fast. Thus causing lack of time and lack of focus, which recursively make situation even worse.

Listed application didn't use the concept of limited resource. One of the ways to illustrate the limited resource was introduced by popular games: bubbles and tetris.

Using a mobile screen as limitation to available tasks, it is possible to construct the following abstraction:

- The vertical axis represents time available till deadline
- The space represents required focus and illustrates which tasks require immediate action

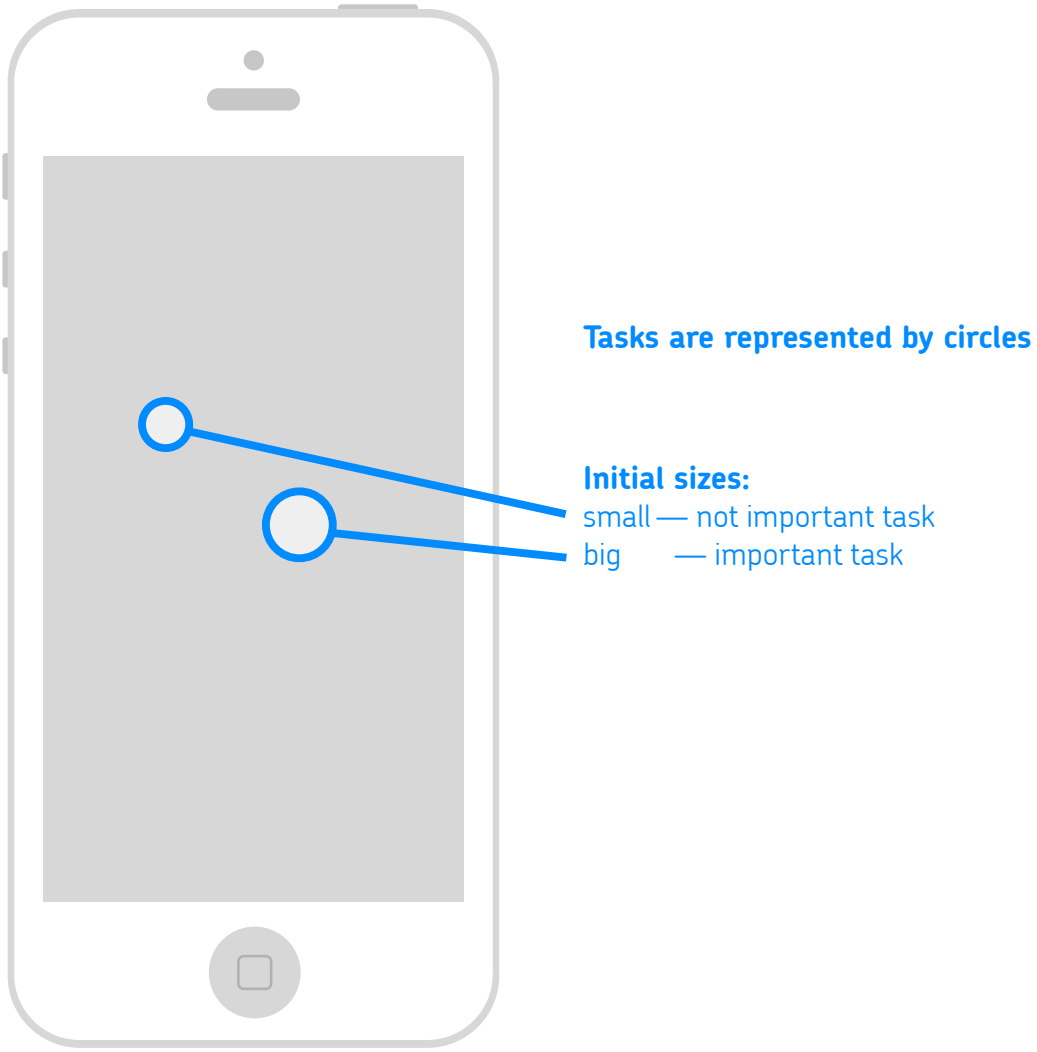

Figure 5.2: In Happy Perform tasks are represented by circles

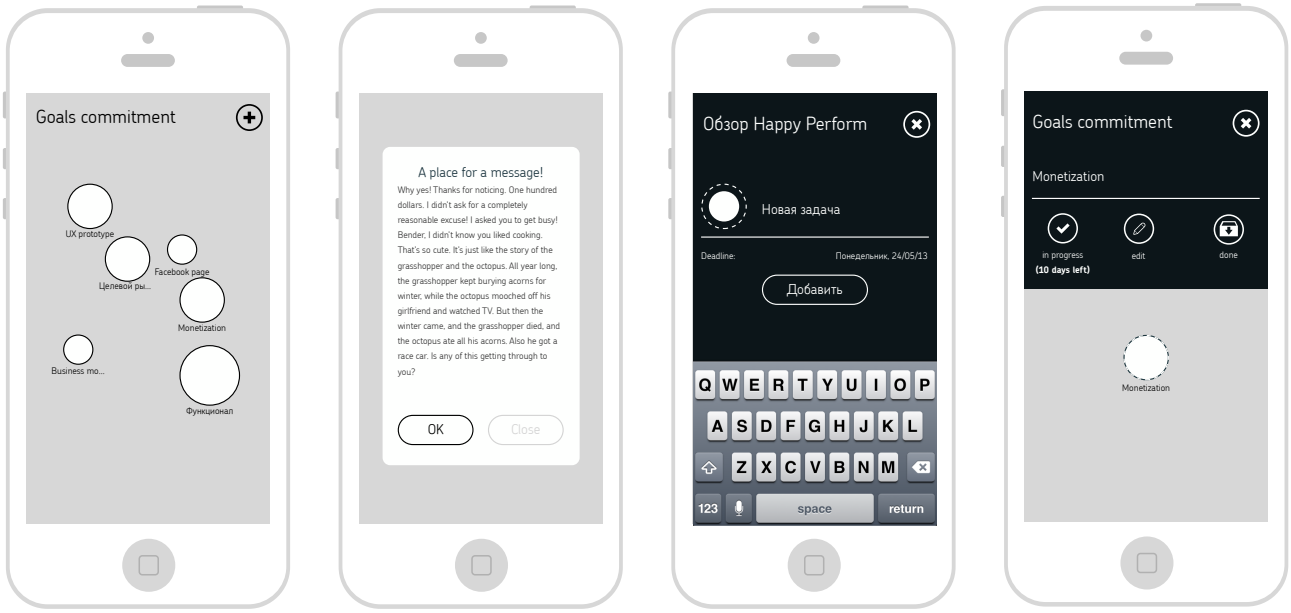

Figure 5.3: Happy Perform: tasks list, message bubble, adding a task, task options

- Each can be represented as a circle, initial (normal) size of the circle is determined by its importance (important or not)
- Space can also group related tasks

5.2 Game mechanics in use

In the case of application there are two main dynamics used:

1. Appointment Dynamic, which requires to return at a predefined time to mark a task as “in progress” or “complete”.
2. Using disincentives to illustrate limited space: if there is not enough space, it is impossible to add new task

5.3 Application workflow

Let’s assume a person have a number of tasks. It is possible to classify them in a following way:

1. recurring tasks
2. tasks with deadlines set too far (e.g.: buy a flat)
3. tasks with deadlines not yet defined
4. tasks with defined start date (e.g.:tuesday, movie)

5. *"frogs", or tasks that are worst or most important (they can be recurring, or anything else)*

Those can be grouped into two categories: continuous and instant. Continuous tasks require recurring "in progress" mark. Instant tasks don't have deadlines.

Let's fill in those tasks into "Happy Perform" application. "Frogs" will get an 'important' status and a bigger circle size.

Deadlines are very important. Also setting a KPI (key performance indicator) to a task is effective: it gives a required feedback. The universal KPI in the case of "Happy Perform" is the size of the circle and position on the screen: tasks start from the top with a certain speed (that is defined with the number of days till deadline) and move towards bottom of the screen, at the day of deadline they will reach the bottom.

Instant tasks (that should be done 'today') appear at the bottom of the screen initially.

Each day a person should mark tasks he did as "in progress" or they will start to grow exponentially. An oversized task start to shrink to initial size if marked as "in progress". Tasks on the bottom will grow even if mark as "in progress", but slower.

There is no mechanism to manage recurring tasks and defined start date tasks in the first version of application.

This concept will be produced and tested as a part of a complex productivity solution.

Conclusion

The objective for this dissertation was to explore modern project management methodologies and gamification theory in order to understand elements affecting performance of project participants and provide guidelines in order to increase their effectiveness.

According to a study, conducted by Dean Spitzer [1], as many as 50 percent of workers said they only put enough effort into their work to hold onto their jobs. And 84 percent said they could work better – if they wanted to. This indicates a problem of ineffectiveness of current productivity solutions.

Research of modern project management techniques led to the common feature discovery: all of them emerged as a trial to reduce complexity of a system in order to determine the cause and effect relations between its elements. That is focus on feedback.

The study of motivation and psychological mechanisms behind it sets a solid foundation to build effective tools and organisational structures.

System thinking approach allows to remember the relations between project management methodologies and personas involved into a project including their motivation systems. This allowed to prototype a mobile productivity application, that uses a notion of “flow” and effective feedback mechanisms.

Bibliography

[1] Dean Spitzer. *Supermotivation*. Soundview Executive Book Summaries.

[2] Dan Brandon. *Project Management for Modern Information Systems*. IRM Press, 2006.

[3] Carl Chatfield. *A short course in project management*, http://office.microsoft.com/en-us/project-help/a-short-course-in-project-management-HA010235482.aspx. Microsoft.

[4] Sebastian Nokes. *The Definitive Guide to Project Management*. ISBN 978-0-273-71097-4. Financial Times, Prentice Hall, 2007.

[5] Agile manifesto, http://www.agilemanifesto.org/.

[6] Kent Beck. Principles behind the agile manifesto, http://www.agilemanifesto.org/principles.html, 2010.

[7] W. Scott Ambler. Supersize me, http://www.drdobbs.com/184415491, 2006.

[8] Robert Schaaf. Agility xl, http://www.sstc-online.org/Proceedings/2007/pdfs/RJS1722.pdf.

[9] The scrum guide, http://scrum.org.

[10] Ellen Gottesdiener. Rad realities: beyond the hype to how rad really works. *Application Development Trends*.

[11] Willard Fazar. *The American Statistition*. April 1959.

[12] Scott Belsky. *Making Ideas Happen*.

[13] Richard Clark. Research-tested team motivation strategies. *Performance Improvement*.

A. Game mechanics list

There are 47 cards, combined by SCVNGR, that describe different game dynamics terms, including mechanisms for gamification of processes.

1. **Achievement**

   Definition: A virtual or physical representation of having accomplished something. These are often viewed as rewards in and of themselves.

   Example: a badge, a level, a reward, points, really anything defined as a reward can be a reward.

2. **Appointment Dynamic**

   Definition: A dynamic in which to succeed, one must return at a predefined time to take some action. Appointment dynamics are often deeply related to interval based reward schedules or avoidance dyanmics.

   Example: Cafe World and Farmville where if you return at a set time to do something you get something good, and if you dont something bad happens.

3. **Avoidance**

   Definition: The act of inducing player behavior not by giving a reward, but by not instituting a punishment. Produces consistent level of activity, timed around the schedule.

   Example: Press a lever every 30 seconds to not get shocked.

4. **Behavioral Contrast**

   Definition: The theory defining how behavior can shift greatly based on changed expectations.

   Example: A monkey presses a lever and is given lettuce. The monkey is happy and continues to press the lever. Then it gets a grape one time. The monkey is delighted. The next time it presses the lever it gets lettuce again. Rather than being happy, as it was before, it goes ballistic throwing

the lettuce at the experimenter. (In some experiments, a second monkey is placed in the cage, but tied to a rope so it cant access the lettuce or lever. After the grape reward is removed, the first monkey beats up the second monkey even though it obviously had nothing to do with the removal. The anger is truly irrational.)

5. **Behavioral Momentum**

   Definition: The tendency of players to keep doing what they have been doing.

   Example: From Jesse Schell's awesome Dice talk: I have spent ten hours playing Farmville. I am a smart person and wouldnt spend 10 hours on something unless it was useful. Therefore this must be useful, so I can keep doing it.

6. **Blissful Productivity**

   Definition: The idea that playing in a game makes you happier working hard, than you would be relaxing. Essentially, were optimized as human beings by working hard, and doing meaningful and rewarding work.

   Example: From Jane McGonicals Ted Talk wherein she discusses how World of Warcraft players play on average 22 hours / week (a part time job), often after a full days work. Theyre willing to work hard, perhaps harder than in real life, because of their blissful productivity in the game world.

7. **Cascading Information Theory**

   Definition: The theory that information should be released in the minimum possible snippets to gain the appropriate level of understanding at each point during a game narrative.

   Example: showing basic actions first, unlocking more as you progress through levels. Making building on SCVNGR a simple but staged process to avoid information overload.

8. **Chain Schedules**

Definition: the practice of linking a reward to a series of contingencies. Players tend to treat these as simply the individual contingencies. Unlocking one step in the contingency is often viewed as an individual reward by the player.

Example: Kill 10 orcs to get into the dragons cave, every 30 minutes the dragon appears.

9. **Communal Discovery**

   Definition: The game dynamic wherein an entire community is rallied to work together to solve a riddle, a problem or a challenge. Immensely viral and very fun.

   Example: DARPA balloon challenge, the cottage industries that appear around McDonalds monopoly to find Boardwalk

10. **Companion Gaming**

    Definition: Games that can be played across multiple platforms

    Example: Games that be played on iphone, facebook, xbox with completely seamless cross platform gameplay.

11. **Contingency**

    Definition: The problem that the player must overcome in the three part paradigm of reward schedules.

    Example: 10 orcs block your path

12. **Countdown**

    Definition: The dynamic in which players are only given a certain amount of time to do something. This will create an activity graph that causes increased initial activity increasing frenetically until time runs out, which is a forced extinction.

    Example: Bejeweled Blitz with 30 seconds to get as many points as you can. Bonus rounds. Timed levels

13. **Cross Situational Leader-boards**

Definition: This occurs when one ranking mechanism is applied across multiple (unequal and isolated) gaming scenarios. Players often perceive that these ranking scenarios are unfair as not all players were presented with an equal opportunity to win.

Example: Players are arbitrarily sent into one of three paths. The winner is determined by the top scorer overall (i.e. across the paths). Since the players can only do one path (and can't pick), they will perceive inequity in the game scenario and get upset.

14. **Disincentives**

    Definition: a game element that uses a penalty (or altered situation) to induce behavioral shift

    Example: losing health points, amazons checkout line removing all links to tunnel the buyer to purchase, speeding traps

15. **Endless Games**

    Definition: Games that do not have an explicit end. Most applicable to casual games that can refresh their content or games where a static (but positive) state is a reward of its own.

    Example: Farmville (static state is its own victory), SCVNGR (challenges constantly are being built by the community to refresh content)

16. **Envy**

    Definition: The desire to have what others have. In order for this to be effective seeing what other people have (voyeurism) must be employed.

    Example: my friend has this item and I want it!

17. **Epic Meaning**

    Definition: players will be highly motivated if they believe they are working to achieve something great, something awe-inspiring, something bigger than themselves.

    Example: From Jane McGonicals Ted Talk where she discusses Warcrafts ongoing story line and epic meaning that involves each individual

has motivated players to participate outside the game and create the second largest wiki in the world to help them achieve their individual quests and collectively their epic meanings.

18. **Extinction**

    Definition: Extinction is the term used to refer to the action of stopping providing a reward. This tends to create anger in players as they feel betrayed by no longer receiving the reward they have come to expect. It generally induces negative behavioral momentum.

    Example: killing 10 orcs no longer gets you a level up

19. **Fixed Interval Reward Schedules**

    Definition: Fixed interval schedules provide a reward after a fixed amount of time, say 30 minutes. This tends to create a low engagement after a reward, and then gradually increasing activity until a reward is given, followed by another lull in engagement.

    Example: Farmville, wait 30 minutes, crops have appeared

20. **Fixed Ratio Reward Schedule**

    Definition: A fixed ratio schedule provides rewards after a fixed number of actions. This creates cyclical nadirs of engagement (because the first action will not create any reward so incentive is low) and then bursts of activity as the reward gets closer and closer.

    Example: kill 20 ships, get a level up, visit five locations, get a badge

21. **Free Lunch**

    Definition: A dynamic in which a player feels that they are getting something for free due to someone else having done work. Its critical that work is perceived to have been done (just not by the player in question) to avoid breaching trust in the scenario. The player must feel that theyve "lucked" into something.

    Example: Groupon. By virtue of 100 other people having bought the deal, you get it for cheap. There is no sketchiness b/c you recognize work

has been done (100 people are spending money) but you yourself didnt have to do it.

22. **Fun Once, Fun Always**

    Definition: The concept that an action in enjoyable to repeat all the time. Generally this has to do with simple actions. There is often also a limitation to the total level of enjoyment of the action.

    Example: the theory behind the check-in everywhere and the check-in and the default challenges on SCVNGR.

23. **Interval Reward Schedules**

    Definition: Interval based reward schedules provide a reward after a certain amount of time. There are two flavors: variable and fixed.

    Example: wait N minutes, collect rent

24. **Lottery**

    Definition: A game dynamic in which the winner is determined solely by chance. This creates a high level of anticipation. The fairness is often suspect, however winners will generally continue to play indefinitely while losers will quickly abandon the game, despite the random nature of the distinction between the two.

    Example: many forms of gambling, scratch tickets.

25. **Loyalty**

    Definition: The concept of feeling a positive sustained connection to an entity leading to a feeling of partial ownership. Often reinforced with a visual representation.

    Example: fealty in WOW, achieving status at physical places (mayorship, being on the wall of favorite customers)

26. **Meta Game**

    Definition: a game which exists layered within another game. These generally are discovered rather than explained (lest they cause confusion) and tend to appeal to 2% of the total gameplaying audience. They

are dangerous as they can induce confusion (if made too overt) but are powerful as theyre greatly satisfying to those who find them.

Example: hidden questions / achievements within world of warcraft that require you to do special (and hard to discover) activities as you go through other quests

27. **Micro Leader-boards**

    Definition: The rankings of all individuals in a micro-set. Often great for distributed game dynamics where you want many micro-competitions or desire to induce loyalty.

    Example: Be the top scorers at Joes bar this week and get a free appetizer

28. **Modifiers**

    Definition: An item that when used affects other actions. Generally modifiers are earned after having completed a series of challenges or core functions.

    Example: A X2 modifier that doubles the points on the next action you take.

29. **Moral Hazard of Game Play**

    Definition: The risk that by rewarding people manipulatively in a game you remove the actual moral value of the action and replace it with an ersatz game-based reward. The risk that by providing too many incentives to take an action, the incentive of actually enjoying the action taken is lost. The corollary to this is that if the points or rewards are taken away, then the person loses all motivation to take the (initially fun on its own) action.

    Example: Paraphrased from Jesse Schell "If I give you points every time you brush your teeth, youll stop brushing your teeth b/c its good for you and then only do it for the points. If the points stop flowing, your teeth will decay."

30. **Ownership**

Definition: The act of controlling something, having it be 'your' property.

Example: Ownership is interesting on a number of levels, from taking over places, to controlling a slot, to simply owning popularity by having a digital representation of many friends.

31. **Pride**

    Definition: the feeling of ownership and joy at an accomplishment

    Example: I have ten badges. I own them. They are mine. There are many like them, but these are mine. Hooray.

32. **Privacy**

    Definition: The concept that certain information is private, not for public distribution. This can be a demotivator (I wont take an action because I dont want to share this) or a motivator (by sharing this I reinforce my own actions).

    Example: Scales the publish your daily weight onto Twitter (these are real and are proven positive motivator for staying on your diet). Or having your location publicly broadcast anytime you do anything (which is invasive and can should be avoided).

33. **Progression Dynamic**

    Definition: a dynamic in which success is granularly displayed and measured through the process of completing itemized tasks.

    Example: a progress bar, leveling up from paladin level 1 to paladin level 60

34. **Ratio Reward Schedules**

    Definition: Ratio schedules provide a reward after a number of actions. There are two flavors: variable and fixed.

    Example: kill 10 orcs, get a power up.

35. **Real-time v. Delayed Mechanics**

Definition: Realtime information flow is uninhibited by delay. Delayed information is only released after a certain interval.

Example: Realtime scores cause instant reaction (gratification or demotivation). Delayed causes ambiguity which can incent more action due to the lack of certainty of ranking.

36. **Reinforcer**

    Definition: The reward given if the expected action is carried out in the three part paradigm of reward schedules.

    Example: receiving a level up after killing 10 orcs.

37. **Response**

    Definition: The expected action from the player in the three part paradigm of reward schedules.

    Example: the player takes the action to kill 10 orcs

38. **Reward Schedules**

    Definition: the timeframe and delivery mechanisms through which rewards (points, prizes, level ups) are delivered. Three main parts exist in a reward schedule; contingency, response and reinforcer.

    Example: getting a level up for killing 10 orcs, clearing a row in Tetris, getting fresh crops in Farmville

39. **Rolling Physical Goods**

    Definition: A physical good (one with real value) that can be won by anyone on an ongoing basis as long as they meet some characteristic. However, that characteristic rolls from player to player.

    Example: top scorer deals, mayor deals

40. **Shell Game**

    Definition: a game in which the player is presented with the illusion of choice but is actually in a situation that guides them to the desired outcome of the operator.

    Example: 3 Card Monty, lotteries, gambling

41. **Social Fabric of Games**

    Definition: the idea that people like one another better after theyve played games with them, have a higher level of trust and a great willingness to work together.

    Example: From Jane McGonicgals TED talk where she suggests that it takes a lot of trust to play a game with someone because you need them to spend their time with you, play by the same rules, shoot for the same goals.

42. **Status**

    Definition: The rank or level of a player. Players are often motivated by trying to reach a higher level or status.

    Example: white paladin level 20 in WOW.

43. **Urgent Optimism**

    Definition: Extreme self motivation. The desire to act immediately to tackle an obstacle combined with the belief that we have a reasonable hope of success.

    Example: From Jane McGonicals TED talk. The idea that in proper games an "epic win" or just "win" is possible and therefore always worth acting for.

44. **Variable Interval Reward Schedules**

    Definition: Variable interval reward schedules provide a reward after a roughly consistent amount of time. This tends to create a reasonably high level of activity over time, as the player could receive a reward at any time but never the burst as created under a fixed schedule. This system is also more immune to the nadir right after the receiving of a reward, but also lacks the zenith of activity before a reward in unlocked due to high levels of ambiguity.

    Example: Wait roughly 30 minutes, a new weapon appears. Check back as often as you want but that wont speed it up. Generally players are bad at realizing that.

45. **Variable Ratio Reward Schedule**

    Definition: A variable ratio reward schedule provides rewards after a roughly consistent but unknown amount of actions. This creates a relatively high consistent rate of activity (as there could always be a reward after the next action) with a slight increase as the expected reward threshold is reached, but never the huge burst of a fixed ratio schedule. Its also more immune to nadirs in engagement after a reward is acheived.

    Example: kill something like 20 ships, get a level up. Visit a couple locations (roughly five) get a badge

46. **Viral Game Mechanics**

    Definition: A game element that requires multiple people to play (or that can be played better with multiple people)

    Example: Farmville making you more successful in the game if you invite your friends, the social check-in

47. **Virtual Items**

    Definition: Digital prizes, rewards, objects found or taken within the course of a game. Often these can be traded or given away.

    Example: Gowallas items, Facebook gifts, badges